PATENT A

Case Docket No. MNFRAME.002A1
Date: October 1, 1997
Page 1

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON, D.C. 20231

ATTENTION: APPLICATION BRANCH

Sir:

Transmitted herewith for filing is the patent application of

Inventor(s): **Tahir Q. Sheikh, Karl S. Johnson and Ken Nguyen**

For: **SYSTEM ARCHITECTURE FOR REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT**

Enclosed are:

(X) Seven (7) sheet(s) of informal drawings.

(X) **UNSIGNED** declaration and power of attorney enclosed.

(X) Appendix A attached to specification.

(X) Appendix B in 21 pages.

(X) Return prepaid postcard.

**CLAIMS AS FILED**

| FOR | NUMBER FILED | NUMBER EXTRA | RATE | FEE |
|---|---|---|---|---|
| Basic Fee | | | $790 | $790 |
| Total Claims | 21 - 20 = | 1 × | $22 | $22 |
| Independent Claims | 3 - 3 = | 0 × | $82 | $0 |
| If application contains any multiple dependent claims(s), then add | | | $270 | $0 |
| | | TOTAL FILING FEE | | $812 |

(X) The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Account No. 11-1410. A duplicate copy of this sheet is enclosed.

(X) A check in the amount of $812 to cover the filing fee is enclosed.

John M. Carson
Registration No. 34,303
Attorney of Record

RJS-3390:cd
100197

PATENT

Case Docket No. **MNFRAME.002A1**
Date: October 1, 1997
Page 1

**ASSISTANT COMMISSIONER FOR PATENTS**
**WASHINGTON, D.C. 20231**

ATTENTION: APPLICATION BRANCH

Sir:

Transmitted herewith for filing is the patent application of

Inventor(s): **Tahir Q. Sheikh, Karl S. Johnson and Ken Nguyen**

For: **SYSTEM ARCHITECTURE FOR REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT**

Enclosed are:

(X) Seven (7) sheet(s) of informal drawings.

(X) **UNSIGNED** declaration and power of attorney enclosed.

(X) Appendix A attached to specification.

(X) Appendix B in 21 pages.

(X) Return prepaid postcard.

**CLAIMS AS FILED**

| FOR | NUMBER FILED | | NUMBER EXTRA | | RATE | FEE |
|---|---|---|---|---|---|---|
| Basic Fee | | | | | $790 | $790 |
| Total Claims | 21 | - 20 = | 1 | × | $22 | $22 |
| Independent Claims | 3 | - 3 = | 0 | × | $82 | $0 |
| If application contains any multiple dependent claims(s), then add | | | | | $270 | $0 |
| | | | **TOTAL FILING FEE** | | | $812 |

(X) The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Account No. 11-1410. A duplicate copy of this sheet is enclosed.

(X) A check in the amount of $812 to cover the filing fee is enclosed.

John M. Carson
Registration No. 34,303
Attorney of Record

RJS-3390:cd
100197

## DECLARATION AND POWER OF ATTORNEY - USA PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name;

I believe I am an original, first and joint inventor of the subject matter which is claimed and for which a patent is sought on the invention entitled SYSTEM ARCHITECTURE FOR REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT; the specification of which is attached hereto;

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above;

I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, Code of Federal Regulations, § 1.56;

I hereby claim the benefit under Title 35, United States Codes § 119(e) of any United States provisional application(s) listed below.

| | |
|---|---|
| **Application No.: 60/046,397** | **Filing Date: May 13, 1997** |
| **Application No.: 60/047,016** | **Filing Date: May 13, 1997** |
| **Application No.: 60/046,416** | **Filing Date: May 13, 1997** |

POWER OF ATTORNEY: I hereby appoint the following attorneys and/or agents to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith (if this application is assigned, I acknowledge that the appointed individuals do not represent me, and that instead they represent the assignee): Louis J. Knobbe, Registration No. 18,780; Don W. Martens, Registration No. 21,107; Gordon H. Olson, Registration No. 20,319; James B. Bear, Registration No. 25,221; Darrell L. Olson, Registration No. 28,247; William B. Bunker, Registration No. 29,365; William H. Nieman, Registration No. 30,201; Lowell Anderson, Registration No. 30,990; Arthur S. Rose, Registration No. 28,038; James F. Lesniak, Registration No. 25,240; Ned A. Israelsen, Registration No. 29,655; Drew S. Hamilton, Registration No. 29,801; Jerry T. Sewell, Registration No. 31,567; John B. Sganga, Jr., Registration No. 31,302; Edward A. Schlatter, Registration No. 32,297; Gerard von Hoffmann, Registration No. 33,043; Joseph R. Re, Registration No. 31,291; John M. Carson, Registration No. 34,303; Andrew H. Simpson, Registration No. 31,469; Daniel E. Altman, Registration No. 34,115; Anita M. Kirkpatrick, Registration No. 32,617; Ernest A. Beutler, Registration No. 19,901; Vito A. Canuso, Registration No. 35,471; William H. Shreve, Registration No. 35,678; Stephen C. Jensen, Registration No. 35,556; Steven J. Nataupsky, Registration No. 37,688; Michael H. Trenholm, Registration No. 37,743; Craig S. Summers, Registration No. 31,430; AnneMarie Kaiser, Registration No. 37,649; Brenton R. Babcock, Registration No. 39,592; Nancy Ways Vensko, Registration No. 36,298; Jonathan A. Barney, Registration No. 34,292;

Ronald J. Schoenbaum, Registration No. 38,297; Richard C. Gilmore, Registration No. 37,335; John R. King, Registration No. 34,362; William S. Reimus, Registration No. 38,279; Christine A. Gritzmacher, Registration No. 40,627; John P. Giezentanner, Registration No. 39,993; Adeel S. Akhtar, Registration No. 41,394; Frederick S. Berretta, Registration No. 38,004; Thomas R. Arno, Registration No. 40,490; David N. Weiss, Registration No. 41,371; James T. Hagler, Registration No. 40,631; Dan Hart, Registration No. 40,637; Lori L. Yamato, Registration No. 40,881, Moses Mares, Registration No. 40,716; Stephen M. Lobbin, Registration No. 41,159; Richard Kim, Registration No. 40,046; Robert F. Gazdzinski, Registration No. 39,990; R. Scott Weide, Registration No. 37,755; Katherine W. White, Registration No. 37,470; Richard E. Campbell, Registration No. 34,790; Raimond J. Salenieks, Registration No. 37,924; Renée E. Canuso, Registration No. 36,657; Michael L. Fuller, Registration No. 36,516; Neil S. Bartfeld, Registration No. 39,901; and Daniel E. Johnson, Registration No. 37,033.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful, false statements may jeopardize the validity of the application or any patent issued thereon.

---

Full name of first inventor: **Tahir Q. Sheikh**

Inventor's signature Unsigned

Date 10/1/97

Residence: **Fremont, California**

Citizenship: **India**

Post Office Address: **3680 Norfolk Road, Fremont, California 94538**

Full name of second inventor: **Karl S. Johnson**

Inventor's signature Unsigned

Date 10/1/97

Residence: **Palo Alto, California**

Citizenship: **United States**

Post Office Address: **544 Tennyson Avenue, Palo Alto, California 94301**

Full name of third inventor: **Ken Nguyen**

Inventor's signature Unsigned

Date 10/1/97

Residence: **San Jose, California**

Citizenship: **United States**

Post Office Address: **1160 Formosa Ridge Drive, San Jose, California 95127**

Send Correspondence To:
KNOBBE, MARTENS, OLSON & BEAR, LLP
620 Newport Center Drive
Sixteenth Floor
Newport Beach, CA 92660-8016

Direct Telephone Calls To:
John M. Carson

(619) 235-8550

RJS-3386:cd
100197

INTELLECTUAL PROPERTY LAW

KNOBBE, MARTENS, OLSON & BEAR

A LIMITED LIABILITY PARTNERSHIP INCLUDING
PROFESSIONAL CORPORATIONS

PATENT, TRADEMARK AND COPYRIGHT CAUSES

KOLL CENTER SAN DIEGO
501 WEST BROADWAY
SUITE 1400
SAN DIEGO, CALIFORNIA 92101-3505
(619) 235-8550
FAX (619) 235-0176
INTERNET: WWW.KMOB.COM

Assistant Commissioner for Patents
Washington, D.C. 20231

## CERTIFICATE OF MAILING BY "EXPRESS MAIL"

| | | |
|---|---|---|
| **Attorney Docket No.** | : | MNFRAME.002A1 |
| **Applicant(s)** | : | SHEIKH, T.Q. et al. |
| **For** | : | SYSTEM ARCHITECTURE FOR REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT |
| **Attorney** | : | John M. Carson |
| **"Express Mail" Mailing Label No.** | : | EM 295 372 291 US |
| **Date of Deposit** | : | October 1, 1997 |

I hereby certify that the accompanying

Transmittal in Duplicate; Specification in 30 pages; Appendix A in 5 pages; Appendix B in 11 pages; 7 sheets of drawings; **UNSIGNED** Declaration and Power of Attorney in 3 pages; Check for Filing Fee; Return Prepaid Postcard

are being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and are addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.

John J Riedel

John J. Riedel

RJS-3395:bb
100197

MNFRAME.002A1 PATENT

# SYSTEM ARCHITECTURE FOR REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT

## Related Applications

The subject matter of U.S. Patent Application entitled "Method of Remote Access and Control of Environmental Management," filed on October 1, 1997, Application No. _________, and having attorney Docket No. MNFRAME.002A2 is related to this application.

## Priority Claim

The benefit under 35 U.S.C. § 119(e) of the following U.S. provisional application(s) is hereby claimed:

| Title | Application No. | Filing Date |
|---|---|---|
| "Remote Access and Control of Environmental Management System" | 60/046,397 | May 13, 1997 |
| "Hardware and Software Architecture for Inter-Connecting an Environmental Management System with a Remote Interface" | 60/047,016 | May 13, 1997 |
| "Self Management Protocol for a Fly-By-Wire Service Processor" | 60/046,416 | May 13, 1997 |

## Appendices

Appendix A, which forms a part of this disclosure, is a list of commonly owned copending U.S. patent applications. Each one of the applications listed in Appendix A is hereby incorporated herein in its entirety by reference thereto.

Appendix B, which forms part of this disclosure, is a copy of the U.S. provisional patent application filed May 13, 1997, entitled "Remote Access and Control of Environmental Management System" and assigned Application No. 60/046,397. Page 1, line 6 of the provisional application has been changed from the original to positively recite that the entire provisional application, including the attached documents, forms part of this disclosure.

## Copyright Rights

A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent files or records, but otherwise reserves all copyright rights whatsoever.

## Background of the Invention

### Field of the Invention

The invention relates to fault tolerant computer systems. More specifically, the invention is directed to a system for providing remote access and control of server environmental management.

### Description of the Related Technology

As enterprise-class servers become more powerful and more capable, they are also becoming increasingly sophisticated and complex. For many companies, these changes lead to concerns over server reliability and manageability, particularly in light of the increasingly critical role of server-based applications. While in the past many systems administrators were comfortable with all of the various components that made up a standards-based network server, today's generation of servers can appear as an incomprehensible, unmanageable black box. Without visibility into the underlying behavior of the system, the administrator must "fly blind." Too often the only indicators the network manager has on the relative health of a particular server is whether or not it is running.

It is well-acknowledged that there is a lack of reliability and availability of most standards-based servers. Server downtime, resulting either from hardware or software faults or from regular maintenance, continues to be a significant problem. By one estimate, the cost of downtime in mission critical environments has risen to an annual total of $4.0 billion for U.S. businesses, with the average downtime event resulting in a $140 thousand loss in the retail industry and a $450 thousand loss in the securities industry. It has been reported that companies lose as much as $250 thousand in employee productivity for every 1% of computer downtime. With emerging Internet, intranet and collaborative applications taking on more essential business roles every day, the cost of network server downtime will continue to spiral upward.

While hardware fault tolerance is an important element of an overall high availability architecture, it is only one piece of the puzzle. Studies show that a significant percentage of network server downtime is caused by transient faults in the I/O subsystem. These faults may be due, for example, to the device driver, the adapter card firmware, or hardware which does not properly handle concurrent errors, and often causes servers to crash or hang. The result is hours of downtime per failure, while a system administrator discovers the failure takes some action, and manually reboots the server. In many cases, data volumes on hard disk drives become corrupt and must be repaired when the volume is mounted. A dismount-and-mount cycle may result from the lack of "hot pluggability" in current standards-based servers. Diagnosing intermittent errors can be a frustrating and time-consuming process. For a system to deliver consistently high availability, it must be resilient to these types of faults. Accurate and available information about such faults is central to diagnosing the underlying problems and taking corrective action.

Modern fault tolerant systems have the functionality to provide the ambient temperature of a storage device enclosure and the operational status of other components such as the cooling fans and power supply. However, a limitation of these server systems is that they do not contain self-managing processes to correct malfunctions. Also, if a malfunction occurs in a typical server, it relies on the operating system software to report, record and manage recovery of the fault. However, many types of faults will prevent such software from carrying out these tasks. For example, a disk drive failure can prevent recording of the fault in a log file on that disk drive. If the system error caused the system to power down, then the system administrator would never know the source of the error.

Traditional systems are lacking in detail and sophistication when notifying system administrators of system malfunctions. System administrators are in need of a graphical user interface for monitoring the health of a network of servers. Administrators need a simple point-and-click interface to evaluate the health of each server in the network. In addition, existing fault tolerant servers rely upon operating system maintained logs for error recording. These systems are not capable of maintaining information when the operating system is inoperable due to a system malfunction. Existing systems do not have a system log for maintaining information when the main computational processors are inoperable.

Another limitation of the typical fault tolerant system is that the control logic for the diagnostic system is associated with a particular processor. Thus, if the environmental control processor malfunctioned, then all diagnostic activity on the computer would cease. In traditional systems, if a controller dedicated to the fan system failed, then all fan activity could cease resulting in overheating and ultimate failure of the server. What is desired is a way to obtain diagnostic information when the server OS is not operational or even when main power to the server is down.

Existing fault tolerant systems also lack the power to remotely control a particular server, such as powering up and down, resetting, retrieving or updating system status, displaying flight recorder and so forth. Such control of the server is desired even when the server power is down. For example, if the operating system on the remote machine failed, then a system administrator would have to physically go to the remote machine to re-boot the malfunctioning machine before any system information could be obtained or diagnostics could be started.

Therefore, a need exists for improvements in server management which will result in greater reliability and dependability of operation. Server users are in need of a management system by which the users can accurately gauge the health of their system. Users need a high availability system that must not only be resilient to faults, but must allow for maintenance, modification, and growth--without downtime. System users must be able to replace failed components, and add new functionality, such as new network interfaces, disk interface cards and storage, without impacting existing users. As system demands grow, organizations must frequently expand, or scale, their computing infrastructure, adding new processing power, memory, storage and I/O capacity. With demand for 24-hour access to critical, server-based information resources, planned system downtime for system service or expansion has become unacceptable.

## Summary of the Invention

Embodiments of the inventive remote access system provides system administrators with new levels of client/server system availability and management. It gives system administrators and network managers a comprehensive view into the underlying health of the server--in real time, whether on-site or off-site. In the event of a failure, the invention enables the administrator to learn why the system failed, why the system was unable to boot, and to control certain functions of the server from a remote station.

One embodiment of the present invention is a system for external management of a server environment, the system comprising a first computer having a plurality of components; a second computer connected to the first computer; a plurality of microcontrollers that monitor components of the first computer; a bus that interconnects the microcontrollers; and a remote interface circuit that interfaces the microcontrollers with the second computer thereby facilitating the external management of the first computer components.

Another embodiment of the present invention is a system for external management of a computer environment, the system comprising a first computer having a plurality of components; a second computer connected to the first computer; means for monitoring components of the first computer; and means for interfacing the monitoring means with the second computer to facilitate the external management of the first computer components.

Yet another embodiment of the present invention is a remote interface, comprising a microcontroller; a memory connected to the microcontroller; a first port capable of receiving and transmitting monitoring data; and a second port capable of receiving and transmitting monitoring data, wherein the second port includes connectivity to a microcontroller bus.

## Brief Description of the Drawings

Figure 1 is a top level block diagram of microcontroller network components utilized by an embodiment of the present invention.

Figure 2 is a block diagram of the server portion of the microcontroller network shown in Figure 1.

Figure 3 is a block diagram of a remote interface board (RIB) that is part of the microcontroller network shown in Figures 1 and 2.

Figure 4 is a diagram of serial protocol message formats utilized by the RIB shown in Figure 3.

Figures 5a and 5b are a flowchart of a RIB microcontroller that is a part of the microcontroller network shown in Figures 1 and 2.

Figure 6 is a diagram of a modem dialing and answering state machine defined in Figure 5a.

## Detailed Description of the Invention

The following detailed description presents a description of certain specific embodiments of the present invention. In this description, reference is made to the drawings wherein like parts are designated with like numerals throughout.

For convenience, the discussion of the invention is organized into the following principal sections: Introduction, Server System, Microcontroller Network, Remote Interface Board, Remote Interface Serial Protocol, and RIB Microcontroller Operation.

## I. INTRODUCTION

The inventive computer server system and client computer includes a distributed hardware environment management system that is built as a small self-contained network of microcontrollers. Operating independently of the system processor and operating system software, embodiments of present invention uses separate processors for providing information and managing the hardware environment including fans, power supplies and temperature.

Initialization, modification and retrieval of system conditions are performed through utilization of a remote interface by issuing commands to the environmental processors. The system conditions may include system log size, presence of faults in the system log, serial number for each of the environmental processors, serial numbers for each power supply of the system, system identification, system log count, power settings and presence, canister presence, temperature, BUS/CORE speed ratio, fan speeds, settings for fan faults, LCD display, Non-Maskable Interrupt (NMI) request bits, CPU fault summary, FRU status, JTAG enable bit, system log information, remote access password, over-temperature fault, CPU error bits, CPU presence, CPU thermal fault bits, and remote port modem. The aforementioned list of capabilities provided by the present environmental system is not all-inclusive.

The server system and client computer provides mechanisms for the evaluation of the data that the system collects and methods for the diagnosis and repair of server problems in a manner that system errors can be effectively and efficiently managed. The time to evaluate and repair problems is minimized. The server system ensures that the system will not go down, so long as sufficient system resources are available to continue operation, but rather degrade gracefully until the faulty components can be replaced.

## II. SERVER SYSTEM

Referring to Figure 1, a server system 100 with a remote client computer will be described. In a one embodiment, the server system hardware environment 100 may be built around a self-contained network of microcontrollers, such as, for example, a remote interface microcontroller on the remote interface board or circuit 104, a system interface microcontroller 106 and a system recorder microcontroller 110. This distributed service processor network 102 may operate as a fully self-contained subsystem within the server system 100, continuously monitoring and managing the physical environment of the machine (e.g., temperature, voltages, fan status). The microcontroller network 102 continues to operate and provides a system administrator with critical system information, regardless of the operational status of the server 100.

Information collected and analyzed by the microcontroller network 102 can be presented to a system administrator using either SNMP-based system management software (not shown), or using microcontroller network Recovery Manager software 130 through a local connection 121 or a dial-in connection 123. The system management software, which interfaces with the operating system (OS) 108 such as Microsoft Windows NT Version 4.0 or Novell Netware Version 4.11, for example, provides the ability to manage the specific characteristics of the server system, including Hot Plug Peripheral Component Interconnect (PCI), power and cooling status, as well as the ability to handle alerts associated with these features.

The microcontroller network Recovery Manager software 130 allows the system administrator to query the status of the server system 100 through the microcontroller network 102, even when the server is down. Using the microcontroller network remote management capability, a system administrator can use the Recovery Manager 130 to re-start a failed system through a modem connection 123. First, the administrator can remotely view the microcontroller network Flight Recorder, a feature that stores all system messages, status and error reports in a circular Non-Volatile Random Access Memory buffer (NVRAM) 112. Then, after determining the cause of the system problem, the administrator can use microcontroller network "fly by wire" capability to reset the system, as well as to power the system off or on. "Fly by wire" denotes that no switch, indicator or other control is directly connected to the function it monitors or controls, but instead, all the control and monitoring connections are made by the microcontroller network 102.

The remote interface board (RIB) 104 interfaces the server system 100 to an external client computer. The RIB 104 may be internal or external to an enclosure of the server 100. Furthermore, the RIB may be incorporated onto another circuit of the server, such as a system board 150 (Figure 2) or a backplane 152 of the server. The RIB 104 connects to either a local client computer 122 at the same location as the server 100 or to a remote client computer 124 through an optional switch 120. The client computer 122/124 may in one embodiment run either Microsoft Windows 95 or Windows NT Workstation version 4.0 operating system (OS) 132.

The client computer 122/124 could be another server, such as, for example, a backup server. The client computer 122/124 could also be a handheld computer such as, for example, a personal digital assistant (PDA). It is not necessary that Operating System software be running on the client computer 122/124. For example, the client computer 122/124 could be hard-wired for specific tasks, or could have special purpose embedded software.

The processor and RAM requirements of the client computer 122/124 are such as necessary by the OS 132. The serial port of the client computer 122/124 may utilize a type 16550A Universal Asynchronous Receiver Transmitter (UART). The switch 120 facilitates either the local connection 121 or the modem connection 123 at any one time, but allows both types of connections to be connected to the switch. In an another embodiment, either the local connection 121 or the modem connection 123 is connected directly to the RIB 104. The local connection 121 utilizes a readily available null-modem serial cable to connect to the local client computer. The modem connection may utilize a Hayes-compatible server modem 126 and a Hayes-compatible client modem 128. In one embodiment, a model V.34X 33.6K data/fax modem available from Zoom is utilized as the client modem and the server modem. In another embodiment, a Sportster 33.6K data/fax modem available from US Robotics is utilized as the client modem.

The steps of connecting the remote client computer 124 to the server 100 will now be briefly described. The remote interface 104 has a serial port connector 204 (Figure 3) that directly connects with a counterpart serial port connector of the external server modem 126 without the use of a cable. If desired, a serial cable could be used to interconnect the remote interface 104 and the server modem 126. The cable end of an AC to DC power adapter (not shown, for example a 120 Volt AC to 7.5 Volt DC, or a 220V, European or Japanese adapter) is then connected to the DC power connector J2 (220, Figure 3) of the remote interface, while the double-prong end is plugged into a 120 Volt AC wall outlet. One end of an RJ-45 parallel-wire data cable 103 is then plugged into an RJ-45 jack (226, Figure 3) on the remote interface 104, while the other end is plugged into a RJ-45 Recovery Manager jack on the server 100. The RJ-45 jack on the server then connects to the microcontroller network 102. The server modem 126 is then connected to a communications network 127 using an appropriate connector. The communications network 127 may be a public switched telephone network, although other modem types and communication networks are envisioned. For example, if cable modems are used for the server modem 126 and client modem 128, the communications network can be a cable television network. As another example, satellite modulator/demodulators can be used in conjunction with a satellite network.

In another embodiment, the server modem to client modem connection may be implemented by an Internet connection utilizing the well known TCP/IP protocol. Any of several Internet access devices, such as modems or network interface cards, may be utilized. Thus, the communications network 127 may utilize either circuit or packet switching.

At the remote client computer 124, a serial cable (25-pin D-shell) 129 is used to interconnect the client modem 128 and the client computer 124. The client modem 128 is then connected to the communications network 127 using an appropriate connector. Each modem is then plugged into an appropriate power source for the modem, such as an AC outlet. At this time, the Recovery Manager software 130 is loaded into the client computer 124, if not already present, and activated.

The steps of connecting the local client computer 122 to the server 100 are similar, but modems are not necessary. The main difference is that the serial port connector of the remote interface 104 connects to a serial port of the local client computer 122 by the null-modem serial cable 121.

## III. MICROCONTROLLER NETWORK

In one embodiment, the invention is implemented by a network of microcontrollers 102 (Figure 1). The microcontrollers may provide functionality for system control, diagnostic routines, self-maintenance control, and event logging processors. A further description of the microcontrollers and microcontroller network is provided in U.S. Patent Application No. ________________, entitled "Diagnostic and Managing Distributed Processor System".

Referring to Figure 2, in one embodiment of the invention, the network of microcontrollers 102 includes ten processors. One of the purposes of the microcontroller network 102 is to transfer messages to the other components of the server system 100. The processors may include: a System Interface controller 106, a CPU A controller 166, a CPU B controller 168, a System Recorder 110, a Chassis controller 170, a Canister A controller 172, a Canister B controller 174, a Canister C controller 176, a Canister D controller 178 and a Remote Interface controller 200. The Remote Interface controller 200 is located on the RIB 104 (Figure 1) which is part of the server system 100, but may preferably be external to the server enclosure. The System Interface controller 106, the CPU A controller 166 and the CPU B controller 168 are located on thr system board 150 in the server 100. Also located on the system board are one or more central processing units (CPUs) or microprocessors 164 and an Industry Standard Architecture (ISA) bus 162 that connects to the System Interface Controller 106. Of course, other buses such as PCI, EISA and Microchannel may be used. The CPU 164 may be any conventional general purpose single-chip or multi-chip microprocessor such as a Pentium®, Pentium® Pro or Pentium® II processor available from Intel Corporation, a SPARC processor available from Sun Microsystems, a MIPS® processor available from Silicon Graphics, Inc., a Power PC® processor available from Motorola, or an ALPHA® processor available from Digital Equipment Corporation. In addition, the CPU 164 may be any conventional special purpose microprocessor such as a digital signal processor or a graphics processor.

The System Recorder 110 and Chassis controller 170, along with the NVRAM 112 that connects to the System Recorder 110, may be located on the backplane 152 of the server 100. The System Recorder 110 and Chassis controller 170 are typically the first microcontrollers to power up when server power is applied. The System Recorder 110, the Chassis controller 170 and the Remote Interface microcontroller 200 are the three microcontrollers that have a bias 5 volt power supplied to them. If main server power is off, an independent power supply source for the bias 5 volt power is provided by the RIB 104 (Figure 1). The Canister controllers 172-178 are not considered to be part of the backplane 152 because they are located on separate cards and are removable.

Each of the microcontrollers has a unique system identifier or address. The addresses are as follows in Table 1:

**TABLE 1**

| Microcontroller | Address |
|---|---|
| System Interface controller 106 | 10 |
| CPU A controller 166 | 03 |
| CPU B controller 168 | 04 |
| System Recorder 110 | 01 |
| Chassis controller 170 | 02 |
| Canister A controller 172 | 20 |
| Canister B controller 174 | 21 |
| Canister C controller 176 | 22 |
| Canister D controller 178 | 23 |
| Remote Interface controller 200 | 11 |

The microcontrollers may be Microchip Technologies, Inc. PIC processors in one embodiment, although other microcontrollers, such as an 8051 available from Intel, an 8751 available from Atmel, and a P80CL580 microprocessor available from Philips, could be utilized. The PIC16C74 (Chassis controller 170) and PIC16C65 (the other controllers) are members of the PIC16CXX family of CMOS, fully-static, EPROM-based 8-bit microcontrollers. The PIC controllers have 192 bytes of RAM, in addition to program memory, three timer/counters, two capture/compare/Pulse Width Modulation modules and two serial ports. The synchronous serial port is configured as a two-wire Inter-Integrated Circuit ($I^2C$) bus in one embodiment of the invention. The PIC controllers use a Harvard architecture in which program and data are accessed from separate memories. This improves bandwidth over traditional von Neumann architecture processors where program and data are fetched from the same memory. Separating program and data memory further allows instructions to be sized differently than the 8-bit wide data word. Instruction opcodes are 14-bit wide making it possible to have all single word instructions. A 14-bit wide program memory access bus fetches a 14-bit instruction in a single cycle.

In one embodiment of the invention, the microcontrollers communicate through an I$^2$C serial bus, also referred to as a microcontroller bus 160. The document "The I$^2$C Bus and How to Use It" (Philips Semiconductor, 1992) is hereby incorporated by reference. The I$^2$C bus is a bidirectional two-wire bus that may operate at a 400 kbps. However, other bus structures and protocols could be employed in connection with this invention. For example, Apple Computer ADB, Universal Serial Bus, IEEE-1394 (Firewire), IEEE-488 (GPIB), RS-485, or Controller Area Network (CAN) could be utilized as the microcontroller bus. Control on the microcontroller bus is distributed. Each microcontroller can be a sender (a master) or a receiver (a slave) and each is interconnected by this bus. A microcontroller directly controls its own resources, and indirectly controls resources of other microcontrollers on the bus.

Here are some of the features of the I$^2$C-bus:

- Two bus lines are utilized: a serial data line (SDA) and a serial clock line (SCL).
- Each device connected to the bus is software addressable by a unique address and simple master/slave relationships exist at all times; masters can operate as master-transmitters or as master-receivers.
- The bus is a true multi-master bus including collision detection and arbitration to prevent data corruption if two or more masters simultaneously initiate data transfer.
- Serial, 8-bit oriented, bidirectional data transfers can be made at up to 400 kbit/second in the fast mode.

Two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the I$^2$C bus. Each device is recognized by a unique address and can operate as either a transmitter or receiver, depending on the function of the device. For example, a memory device connected to the I$^2$C bus could both receive and transmit data. In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers (see Table 2). A master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave.

**TABLE 2 Definition of I$^2$C-bus terminology**

| Term | Description |
|---|---|
| Transmitter | The device which sends the data to the bus |
| Receiver | The device which receives the data from the bus |
| Master | The device which initiates a transfer, generates clock signals and terminates a transfer |
| Slave | The device addressed by a master |
| Multi-master | More than one master can attempt to control the bus at the same time without corrupting the message |
| Arbitration | Procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted |
| Synchronization | Procedure to synchronize the clock signal of two or more devices |

The I$^2$C-bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. As masters are usually microcontrollers, consider the case of a data transfer between two microcontrollers connected to the I$^2$C-bus. This highlights the master-slave and receiver-transmitter relationships to be found on the I$^2$C-bus. It should be noted that these relationships are not permanent, but depend on the direction of data transfer at that time. The transfer of data would proceed as follows:

1) Suppose microcontroller A wants to send information to microcontroller B:

- microcontroller A (master), addresses microcontroller B (slave);
- microcontroller A (master-transmitter), sends data to microcontroller B (slave-receiver);
- microcontroller A terminates the transfer.

2) If microcontroller A wants to receive information from microcontroller B:

- microcontroller A (master addresses microcontroller B (slave);
- microcontroller A (master-receiver) receives data from microcontroller B (slave-transmitter);
- microcontroller A terminates the transfer.

Even in this situation, the master (microcontroller A) generates the timing and terminates the transfer.

The possibility of connecting more than one microcontroller to the $I^2C$-bus means that more than one master could try to initiate a data transfer at the same time. To avoid the chaos that might ensue from such an event, an arbitration procedure has been developed. This procedure relies on the wired-AND connection of all $I^2C$ interfaces to the $I^2C$-bus.

If two or more masters try to put information onto the bus, the first to produce a 'one' when the other produces a 'zero' will lose the arbitration. The clock signals during arbitration are a synchronized combination of the clocks generated by the masters using the wired-AND connection to the SCL line.

Generation of clock signal on the $I^2C$-bus is the responsibility of master devices. Each master microcontroller generates its own clock signals when transferring data on the bus.

The command, diagnostic, monitoring and history functions of the microcontroller network 102 are accessed using a global network memory model in one embodiment. That is, any function may be queried simply by generating a network "read" request targeted at the function's known global network address. In the same fashion, a function may be exercised simply by "writing" to its global network address. Any microcontroller may initiate read/write activity by sending a message on the $I^2C$ bus to the microcontroller responsible for the function (which can be determined from the known global address of the function). The network memory model includes typing information as part of the memory addressing information.

Using a network global memory model in one embodiment places relatively modest requirements for the $I^2C$ message protocol.

- All messages conform to the $I^2C$ message format including addressing and read/write indication.
- All $I^2C$ messages use seven bit addressing.
- Any controller can originate (be a Master) or respond (be a Slave).
- All message transactions consist of $I^2C$ "Combined format" messages. This is made up of two back-to-back $I^2C$ simple messages with a repeated START condition between (which does not allow for re-arbitrating the bus). The first message is a Write (Master to Slave) and the second message is a Read (Slave to Master).
- Two types of transactions are used: Memory-Read and Memory-Write.
- Sub-Addressing formats vary depending on data type being used.

## IV. REMOTE INTERFACE BOARD

Referring to Figure 3, the remote interface board (RIB) 104, previously shown in Figure 1, will now be described. The RIB is an interface between the microcontroller network 102 (Figure 1) of the server system 100 and an external client computer 122/124. The server system status and commands are passed through the RS232 connector port 204 at the client side of the RIB to the microcontroller network 102 on the server 100, controlled through the on-board PIC16C65 microcontroller 200. Signals in the microcontroller network 102 are transported by the microcontroller bus 160 (Figure 2). In one embodiment, the microcontroller bus 160 utilizes the $I^2C$ bus protocol, previously described. The signals on the microcontroller bus 160 are received from the server 100 by the RIB 104 on the RJ-45 cable 103 and are translated by the PIC16C65 microcontroller 200 into an eight signal RS232 protocol. These RS232 signals are passed through a RS232 line transceiver 202, such as a LT1133A chip available from Linear Technology, with a baud rate capable of reaching the speed of 120 kbaud. A 25 pin D-Sub connector 204 connects to the other side of the line transceiver 202 and provides the point at which either the local client computer 122 or the server modem 126 makes a connection.

The two wire microcontroller bus 160 is brought in from the server 100 and passed to the microcontroller 200 using the RJ-45 cable 103 and RJ-45 connector 226.

A switch 228, such as a QS3126 switch available from Quick Logic, connects to the RJ-45 connector 226 and provides isolation for the data and clock bus signals internal and external to the RIB 104. If the RIB 104 and switch 228 have power, the switch 228 feeds the bus signals through to a microcontroller bus extender 230. Otherwise, if the switch 228 does not have power, the microcontroller bus 160 is isolated from the RIB 104. The bus extender 230 connects between the switch 228 and the microcontroller 200. The bus extender 230 is a buffer providing drive capability for the clock and data signals. In one embodiment, the bus extender 230 is a 82B715 chip available from Philips Semiconductor. Microcontroller 200 Port C, bit 3 is the clocking bit and Port C, bit 4 is the data line.

Communication with the server modem 126 is based on the RS232 protocol. The microcontroller 200 generates the receive and the transmit signals, where the signal levels are transposed to the RS232 levels by the LT1133A line transceiver 202. There are three transmit signals, RTS, SOUT and DTR, which are from Port A, bits 2, 3 and 4 of the microcontroller 200, whereas the five receive signals are from two ports, DCD, DSR from Port C, bits 1 and 0 and SIN, CTS and RI from Port A, bits 5, 0 and 1.

In one embodiment, the 25 pin RS232 pin connector 204 is used instead a 9 pin connector, since this type of connector is more common. All the extra pins are not connected except the pins 1 and 7, where pin 1 is chassis ground and pin 7 is a signal ground.

A static random access memory (SRAM) 208 connects to the microcontroller 200. In one embodiment, the SRAM 208 is a 32k x 8 MT5LC2568 that is available from Micron Technology. The SRAM 208 is also available from other memory manufacturers. An external address register 206, such as an ABT374, available from Texas Instruments is used for latching the higher addressing bits (A8-A14) of the address for the SRAM 208 so as to expand the address to fifteen bits. The SRAM 208 is used to store system status data, system log data from the NVRAM 112 (Figure 1), and other message data for transfer to the external interface port 204 or to a microcontroller on the microcontroller bus 160 (Figure 2).

Port D of the microcontroller 200 is the address port. Port B is the data bus for the bi-directional data interconnect. Port E is for the SRAM enable, output tristate and write control signals. The microcontroller 200 operates at a frequency of 12 MHz.

An Erasable Programmable Read Only Memory (EPROM) 212 is used for storing board serial number identification information for the RIB 104. The serial number memory 212 is signal powered, retaining the charge into a capacitor sourced through the data line. In one embodiment, the serial number memory 212 stores eight sixteen-byte serial/revision numbers (for maintaining the rework/revision history) and is a DS2502 chip available from Dallas Semiconductor. The programming of memory 212 is handled using a jumper applied through an external connector J1 210. The serial number memory 212 connects to the microcontroller 200 at Port C, bit 6 and to the external connector J1 210.

The RIB 104 may be powered through a 7.5 Volt/800mA supply unit that plugs into a connector J2 220. In one embodiment, the supply unit is 120 Volt AC to DC wall adapter. Connector J2 220 feeds a LT1376 high frequency switching regulator 222, available from Linear Technology, which regulates the power source. The regulated power output is used locally by the components on the RIB 104, and 300 mA are sourced to the microcontroller network 102 through a 300 mA fuse 224 and the RJ-45 connector 226. Thus, the output of the regulator 222 provides an alternative source for a bias-powered partition of the microcontroller network 102. The bias-powered partition includes the system recorder 110 (Figure 1), the NVRAM 112 and the Chassis controller 170 (Figure 2) which are resident on the server backplane 152.

## V. REMOTE INTERFACE SERIAL PROTOCOL

The microcontroller network remote interface serial protocol communicates microcontroller network messages across a point-to-point serial link. This link is between the RIB controller 200 that is in communication with the Recovery Manager 130 at the remote client 122/124. This protocol encapsulates microcontroller network messages in a transmission packet to provide error-free communication and link security.

In one embodiment, the remote interface serial protocol uses the concept of byte stuffing. This means that certain byte values in the data stream have a particular meaning. If that byte value is transmitted by the underlying application as data, it must be transmitted as a two-byte sequence.

The bytes that have a special meaning in this protocol are:

| | |
|---|---|
| SOM 306 | Start of a message |
| EOM 316 | End of a message |
| SUB | The next byte in the data stream must be substituted before processing. |
| INT 320 | Event Interrupt |
| Data 312 | An entire microcontroller network message |

As stated above, if any of these byte values occur as data in a message, a two-byte sequence must be substituted for that byte. The sequence is a byte with the value of SUB, followed by a type with the value of the original byte, which is incremented by one. For example, if a SUB byte occurs in a message, it is transmitted as a SUB followed by a byte that has a value of SUB+1.

Referring to Figure 4, the two types of messages 300 used by the remote interface serial protocol will be described.

1. Requests 302, which are sent by remote management (client) computers 122/124 (Figure 1) to the remote interface 104.
2. Responses 304, which are returned to the requester 122/124 by the remote interface 104.

The fields of the messages are defined as follows:

| | |
|---|---|
| SOM 306 | A special data byte value marking the start of a message. |
| EOM 316 | A special data byte value marking the end of a message. |
| Seq.# 308 | A one-byte sequence number, which is incremented on each request. It is stored in the response. |

TYPE 310 One of the following types of requests:

IDENTIFY Requests the remote interface to send back identification information about the system to which it is connected. It also resets the next expected sequence number. Security authorization does not need to be established before the request is issued.

SECURE Establishes secure authorization on the serial link by checking password security data provided in the message with the microcontroller network password.

UNSECURE Clears security authorization on the link and attempts to disconnect it. This requires security authorization to have been previously established.

MESSAGE Passes the data portions of the message to the microcontroller network for execution. The response from the microcontroller network is sent back in the data portion of the response. This requires security authorization to have been previously established.

POLL Queries the status of the remote interface. This request is generally used to determine if an event is pending in the remote interface.

STATUS 318 One of the following response status values:

OK Everything relating to communication with the remote interface is successful.

OK_EVENT Everything relating to communication with the remote interface is successful. In addition, there is one or more events pending in the remote interface.

SEQUENCE The sequence number of the request is neither the current sequence number or retransmission request, nor the next expected sequence number or new request. Sequence numbers may be reset by an IDENTIFY request.

CHECK The check byte in the request message is received incorrectly.

FORMAT Something about the format of the message is incorrect. Most likely, the type field contains an invalid value.

SECURE The message requires that security authorization be in effect, or, if the message has a TYPE value of SECURE, the security check failed.

Check 314 Indicates a message integrity check byte. Currently the value is 256 minus the sum of previous bytes in the message. For example, adding all bytes in the message up to and including the check byte should produce a result of zero (0).

INT 320 A special one-byte message sent by the remote interface when it detects the transition from no events pending to one or more events pending. This message can be used to trigger reading events from the remote interface. Events should be read until the return status changes form OK_EVENT to OK.

## VI. RIB MICROCONTROLLER OPERATION

The remote interface is the bridge to link the microcontroller bus to the outside world via a RS232 serial port through which a client computer can be connected. A message from the remote client side via RS232 usually starts with the "Identify" command which identifies the system name. See the message format associated with Figure 4, above. The "Identify" command should be followed by the "Security" command with a password that is checked against the password stored in the NVRAM 112 (Figure 1). If the passwords match, the remote RS232 link is put in "secure mode" and the remote interface 104 (Figure 1) will now pass any "message" commands on to the microcontroller network bus 160 (Figure 2). Before the remote application program disconnects the link, it should send the "Unsecure" command to take the RS232 link out of "secure mode".

Referring to Figures 5a and 5b, embodiments of the RIB microcontroller process 400 will be described. The process 400 is implemented as a computer program, termed firmware, written in PIC assembly language. The assembled machine code is stored in the microcontroller EPROM where each instruction is fetched for execution by the processor. The EPROM provides 4K x 14 program memory space, all on-chip. Program execution is using the internal memory. Of course, any of a variety of general purpose and special purpose processors could be used and the programming of the process 400 could be in high level code such as C or Java.

Beginning at an initialize PIC state 402, process 400 initializes the variables, stack pointer, and other structures of the RIB microcontroller 200 (Figure 3). Moving to state 404, a return point called "main" is identified in process 400. Proceeding to a decision state 406, process 400 determines if the RS232 port is transmitting data. If so, process 400 moves to state 408 to send a character (one byte) if there is data in the SRAM 208 to be sent out on the RS232 port 204. A process of receiving data via the RS232 port 204 is not shown herein. Receiving data via the port 204 is initiated by the use of an interrupt.

At the completion of state 408, or if decision state 406 evaluates to a false condition, process 400 proceeds to a Check Modem Status function 410 that is implemented as a modem dialing and answering state machine. Function 410 checks the status of the modem 126 for any possible activity. Function 410 will be further described in conjunction with Figure 6. Advancing to a decision state 412, process 400 determines if any server event is pending. Event types include, for example, CPU status change, power status change, canister status change, fan status change, temperature, and operating system timeout. If an event is pending, process 400 proceeds to state 414 and sends an event message to the client computer 122/124 via the RS232 port. If no event is pending, as determined at decision state 412, process 400 continues at a decision state 416. At decision state 416, process 400 checks to see if a RS232 remote message has been received from the client computer 122/124. If not, process 400 moves back to the "main" loop 404, as described above. One reason that a message has not been received yet is that the modem is not yet transmitting.

If a message has been received, as determined at decision state 416, process moves to the appropriate state 420-426 to handle one of four command types: Identify, Secure, Unsecure, and Message. At state 420, process 400 performs the Identify command and identifies the system by responding with the system name retrieved from the System Recorder memory 112 (Figure 1).

At state 422, process 400 performs the Secure command and gets the password with the command and checks it against the password from the NVRAM 112 (Figure 1). If the passwords match, the access right is granted (opens secure mode), otherwise, reject the intent.

At state 424, process 400 performs the Unsecure command and releases the remote access right, i.e., closes secure mode. At the completion of states 420, 422 or 424, process 400 proceeds through off-page connector E 430 to state 438 (Figure 5b).

At state 426 on Figure 5b (through off-page connector D 418), process 400 performs the Message command and gets remote message data from the RIB SRAM 208 (Figure 3). Proceeding to a decision state 432, process 400 determines if this message command is for the remote interface 104. If it is, process 400 executes the internal remote interface function command, such as a Read Revision of the RIB command. If the message command is not for the remote interface, as determined by decision state 432, process 400 moves to state 436 and passes the message command to its destination (external to the remote interface) via the microcontroller bus. This facilitates communication with another microcontroller for a command to read or write information, for example.

At the completion of states 420, 422, 424, 434 or 436, process 400 advances to state 438 and stores the response data for the command into the SRAM 208 (Figure 3) to be sent back to the client computer 122/124. Moving to state 440, process 400 transmit the first byte of data back on the RS232 port 204 to the client computer 122/124. After the byte of data has been transmitted at state 440, process 400 moves back to the "main" loop 404 (on Figure 5a), as described above.

Referring to Figure 6, embodiments of the Check Modem Status function 410 will now be described. Function 410 is implemented as a modem dialing and answering state machine. Several terms useful for understanding of the modem dialing and answering state machine are listed in Table 3 below.

**TABLE 3**

| Modem Term | Meaning |
|---|---|
| CTS | clear to send |
| DCD | data carrier detect |
| DSR | data set ready |
| DTR | data transfer ready |
| RTS | request to send |
| EOS | end of string |
| Protocol | indicates whether RS232 serial data uses the messaging protocol or whether the data is a string of bytes |
| Ring | modem is detecting an incoming ring signal from another modem |
| Local | a connection to a local client computer (no modem used) |
| Modem Mode | modem to modem connection |
| Modem Already Set | modem initialization string has already been sent and completed |

State machine 410 includes nine states, states 470-486. State 470 denotes that the modem is disconnected, DTR and RTS are clear and the protocol is clear. Protocol is clear indicates that no message protocol processing is to occur for bytes on the RS232 link (because it would affect transmitting and receiving of modem control string bytes). The state machine 410 remains at the Modem Disconnect state 470 while CTS is clear OR there have been "n" dialing retries already OR there is no Ring OR DSR is clear. If DSR is set (active), the state machine 410 proceeds to a Local Modem state 486, wherein RTS and DTR are set. The state machine 410 remains at state 486 while DSR is set. Is DSR clears or if Local AND Modem Mode are both set, the state machine 410 returns to Modem Disconnect state 470.

The state machine 410 proceeds to Modem Soft Reset state 472 if a Call Out condition OR a Setup condition is achieved. Call Out is achieved if Modem Mode is set AND Modem Already Set is set AND CTS is set AND there have not been "n" dialing retries already. Setup is achieved if Modem Mode is set AND Modem Already Set is clear AND CTS is set. At Modem Soft Reset state 472, DTR is set and RTS is set. The state machine 410 remains at state 472 while Send String Done is clear, i.e., the modem command string is still being sent to the modem.

The state machine 410 proceeds to Modem Test state 474 when Send String Done is set. The state machine 410 remains at state 474 while Send String Done is clear. The state machine 410 proceeds to Modem Result Code state 476 when Send String Done is set. The state machine 410 remains at state 476 while Modem Result Status Done is clear, i.e., the results status of the modem test at state 474 is not yet available.

The state machine 410 returns to Modem Disconnect state 470 from state 476 if Results Status OK is clear, i.e., the results status is not OK. However, if Results Status OK is set, i.e., the results status is correct, the state machine 410 proceeds to a Modem Setup state 478, wherein Modem Already Set is set. The state machine 410 returns to Modem Disconnect state 470 from state 478 if there have been "n" dialing retries already. However, if there have not been "n" dialing retries already, the state machine 410 proceeds to a Modem Dialing state 480, wherein the modem is dialed.

The state machine 410 remains at state 480 while the previous EOS has not been reached AND two seconds have not passed. The state machine 410 returns to Modem Disconnect state 470 from state 480 if Dial OK is clear, i.e., dialing the modem was not successful. However, if Dial OK is set, i.e., dialing the modem was successful, the state machine 410 proceeds to a Modem Answering state 482. Another path to the Modem Answering state 482 is from the Modem Disconnect state 470 when a Ringing mode is achieved. Ringing mode is achieved if Modem Mode is set AND Modem Already Set is set AND CTS is set AND Ring is set. The state machine 410 remains at state 482 while DSR is clear OR DCD is clear. The state machine 410 returns to Modem Disconnect State 470 from state 482 if DCD is clear and a timeout occurs, i.e., no DCD is set within a timeout period (nobody answers). The state machine 410 proceeds to Remote Modem state 484 when DSR is set AND DCD is set. The modem transfers message data while at this state. When DCD clears, the state machine 410 returns to Modem Disconnect state 470 from state 484 or otherwise remains at state 484.

While the above detailed description has shown, described, and pointed out the fundamental novel features of the invention as applied to various embodiments, it will be understood that various omissions and substitutions and changes in the form and details of the system illustrated may be made by those skilled in the art, without departing from the intent of the invention.

WHAT IS CLAIMED IS:

1. A system for external management of a computer environment, the system comprising:

a first computer having a plurality of components;

a second computer connected to the first computer;

a plurality of microcontrollers that monitor components of the first computer;

a bus that interconnects the microcontrollers; and

a remote interface circuit that interfaces the microcontrollers with the second computer thereby facilitating the external management of the first computer components.

2. The system defined in Claim 1, wherein the remote interface circuit includes an external port for connection to the second computer.

3. The system defined in Claim 1, wherein the second computer is at the same location as the first computer.

4. The system defined in Claim 1, wherein the second computer is at a location remote to the first computer.

5. The system defined in Claim 4, additionally comprising a pair of modems, wherein a first modem connects to the first computer and a second modem connects to the second computer.

6. The system defined in Claim 5, wherein each modem further connects to the public switched telephone network.

7. The system defined in Claim 5, wherein each modem further connects to a cable network.

8. The system defined in Claim 1, wherein the remote interface circuit includes a remote interface microcontroller that connects via the bus to the microcontrollers.

9. The system defined in Claim 1, wherein the remote interface circuit includes a memory for storing results of monitoring the first computer components.

10. The system defined in Claim 1, wherein the remote interface circuit includes a power source independent of the first computer.

11. The system defined in Claim 10, wherein the independent power source includes an AC to DC adapter.

12. The system defined in Claim 1, wherein the second computer is a handheld computer.

13. A system for external management of a computer environment, the system comprising:

a first computer having a plurality of components;

a second computer connected to the first computer;

means for monitoring components of the first computer; and

means for interfacing the monitoring means with the second computer to facilitate the external management of the first computer components.

14. A remote interface, comprising:

a microcontroller;

a memory connected to the microcontroller;

a first port capable of receiving and transmitting monitoring data; and

a second port capable of receiving and transmitting monitoring data, wherein the second port includes connectivity to a microcontroller bus.

15. The remote interface of Claim 14, wherein the microcontroller bus is $I^2C$.

16. The remote interface of Claim 14, wherein the microcontroller bus is Controller Area Network.

17. The remote interface of Claim 14, wherein the monitoring data is stored in the memory.

18. The remote interface of Claim 14, wherein the first port receives a command from an external computer.

19. The remote interface of Claim 18, wherein the command establishes security authorization.

20. The remote interface of Claim 18, wherein the command requests data from the memory.

21. The remote interface of Claim 18, wherein the command sends data to the microcontroller bus.

# SYSTEM ARCHITECTURE FOR REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT

## Abstract of the Disclosure

A fault tolerant system by which individual components of a server are monitored and controlled through independent, programmable microcontrollers interconnected through a microcontroller network. An external agent can control and monitor the microcontrollers by extending the interconnection network beyond the physical server. The extension to the interconnection network converts protocols between media, and directs the microcontrollers and the state managed by the microcontrollers. Intervention of the server operating system software is not required and is not utilized for the access and control operations. A remote interface board provides the interface between the microcontroller network and an external modem that communicates with a remote client computer. The remote interface board also provides for connection to a local client computer.

RJS-3350://ssw/sad4
093097

## Appendix A

## Incorporation by Reference of Commonly Owned Applications

The following patent applications, commonly owned and filed on the same day as the present application are hereby incorporated herein in their entirety by reference thereto:

| Title | Application No. | Attorney Docket No. |
|---|---|---|
| "System Architecture for Remote Access and Control of Environmental Management" | | MNFRAME.002A1 |
| "Method of Remote Access and Control of Environmental Management" | | MNFRAME.002A2 |
| "System for Independent Powering of Diagnostic Processes on a Computer System" | | MNFRAME.002A3 |
| "Method of Independent Powering of Diagnostic Processes on a Computer System" | | MNFRAME.002A4 |
| "Diagnostic and Managing Distributed Processor System" | | MNFRAME.005A1 |
| "Method for Managing a Distributed Processor System" | | MNFRAME.005A2 |
| "System for Mapping Environmental Resources to Memory for Program Access" | | MNFRAME.005A3 |
| "Method for Mapping Environmental Resources to Memory for Program Access" | | MNFRAME.005A4 |
| "Hot Add of Devices Software Architecture" | | MNFRAME.006A1 |
| "Method for The Hot Add of Devices" | | MNFRAME.006A2 |
| "Hot Swap of Devices Software Architecture" | | MNFRAME.006A3 |
| "Method for The Hot Swap of Devices" | | MNFRAME.006A4 |

| Title | Application No. | Attorney Docket No. |
|---|---|---|
| "Method for the Hot Add of a Network Adapter on a System Including a Dynamically Loaded Adapter Driver" | | MNFRAME.006A5 |
| "Method for the Hot Add of a Mass Storage Adapter on a System Including a Statically Loaded Adapter Driver" | | MNFRAME.006A6 |
| "Method for the Hot Add of a Network Adapter on a System Including a Statically Loaded Adapter Driver" | | MNFRAME.006A7 |
| "Method for the Hot Add of a Mass Storage Adapter on a System Including a Dynamically Loaded Adapter Driver" | | MNFRAME.006A8 |
| "Method for the Hot Swap of a Network Adapter on a System Including a Dynamically Loaded Adapter Driver" | | MNFRAME.006A9 |
| "Method for the Hot Swap of a Mass Storage Adapter on a System Including a Statically Loaded Adapter Driver" | | MNFRAME.006A10 |
| "Method for the Hot Swap of a Network Adapter on a System Including a Statically Loaded Adapter Driver" | | MNFRAME.006A11 |
| "Method for the Hot Swap of a Mass Storage Adapter on a System Including a Dynamically Loaded Adapter Driver" | | MNFRAME.006A12 |
| "Method of Performing an Extensive Diagnostic Test in Conjunction with a BIOS Test Routine" | | MNFRAME.008A |
| "Apparatus for Performing an Extensive Diagnostic Test in Conjunction with a BIOS Test Routine" | | MNFRAME.009A |
| "Configuration Management Method for Hot Adding and Hot Replacing Devices" | | MNFRAME.010A |

| Title | Application No. | Attorney Docket No. |
|---|---|---|
| "Configuration Management System for Hot Adding and Hot Replacing Devices" | | MNFRAME.011A |
| "Apparatus for Interfacing Buses" | | MNFRAME.012A |
| "Method for Interfacing Buses" | | MNFRAME.013A |
| "Computer Fan Speed Control Device" | | MNFRAME.016A |
| "Computer Fan Speed Control Method" | | MNFRAME.017A |
| "System for Powering Up and Powering Down a Server" | | MNFRAME.018A |
| "Method of Powering Up and Powering Down a Server" | | MNFRAME.019A |
| "System for Resetting a Server" | | MNFRAME.020A |
| "Method of Resetting a Server" | | MNFRAME.021A |
| "System for Displaying Flight Recorder" | | MNFRAME.022A |
| "Method of Displaying Flight Recorder" | | MNFRAME.023A |
| "Synchronous Communication Interface" | | MNFRAME.024A |
| "Synchronous Communication Emulation" | | MNFRAME.025A |
| "Software System Facilitating the Replacement or Insertion of Devices in a Computer System" | | MNFRAME.026A |
| "Method for Facilitating the Replacement or Insertion of Devices in a Computer System" | | MNFRAME.027A |
| "System Management Graphical User Interface" | | MNFRAME.028A |
| "Display of System Information" | | MNFRAME.029A |
| "Data Management System Supporting Hot Plug Operations on a Computer" | | MNFRAME.030A |

| Title | Application No. | Attorney Docket No. |
|---|---|---|
| "Data Management Method Supporting Hot Plug Operations on a Computer" | | MNFRAME.031A |
| "Alert Configurator and Manager" | | MNFRAME.032A |
| "Managing Computer System Alerts" | | MNFRAME.033A |
| "Computer Fan Speed Control System" | | MNFRAME.034A |
| "Computer Fan Speed Control System Method" | | MNFRAME.035A |
| "Black Box Recorder for Information System Events" | | MNFRAME.036A |
| "Method of Recording Information System Events" | | MNFRAME.037A |
| "Method for Automatically Reporting a System Failure in a Server" | | MNFRAME.040A |
| "System for Automatically Reporting a System Failure in a Server" | | MNFRAME.041A |
| "Expansion of PCI Bus Loading Capacity" | | MNFRAME.042A |
| "Method for Expanding PCI Bus Loading Capacity" | | MNFRAME.043A |
| "System for Displaying System Status" | | MNFRAME.044A |
| "Method of Displaying System Status" | | MNFRAME.045A |
| "Fault Tolerant Computer System" | | MNFRAME.046A |
| "Method for Hot Swapping of Network Components" | | MNFRAME.047A |
| "A Method for Communicating a Software Generated Pulse Waveform Between Two Servers in a Network" | | MNFRAME.048A |
| "A System for Communicating a Software Generated Pulse Waveform Between Two Servers in a Network" | | MNFRAME.049A |
| "Method for Clustering Software Applications" | | MNFRAME.050A |

| Title | Application No. | Attorney Docket No. |
|---|---|---|
| "System for Clustering Software Applications" | | MNFRAME.051A |
| "Method for Automatically Configuring a Server after Hot Add of a Device" | | MNFRAME.052A |
| "System for Automatically Configuring a Server after Hot Add of a Device" | | MNFRAME.053A |
| "Method of Automatically Configuring and Formatting a Computer System and Installing Software" | | MNFRAME.054A |
| "System for Automatically Configuring and Formatting a Computer System and Installing Software" | | MNFRAME.055A |
| "Determining Slot Numbers in a Computer" | | MNFRAME.056A |
| "System for Detecting Errors in a Network" | | MNFRAME.058A |
| "Method of Detecting Errors in a Network" | | MNFRAME.059A |
| "System for Detecting Network Errors" | | MNFRAME.060A |
| "Method of Detecting Network Errors" | | MNFRAME.061A |

RJS-3350:sad
093097

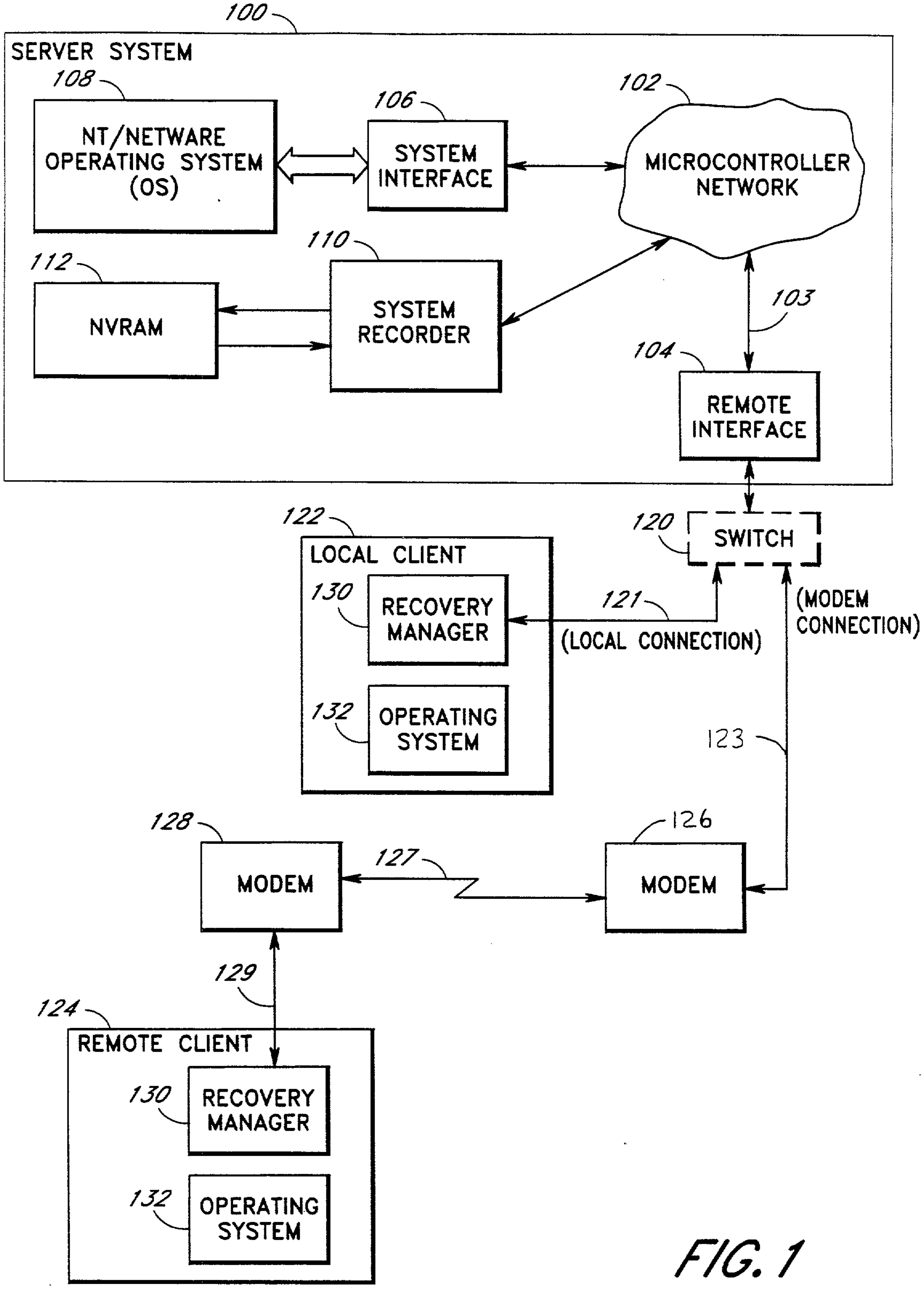

100
SERVER SYSTEM
108
NT/NETWARE OPERATING SYSTEM (OS)
106
SYSTEM INTERFACE
102
MICROCONTROLLER NETWORK
110
SYSTEM RECORDER
112
NVRAM
103
104
REMOTE INTERFACE
122
LOCAL CLIENT
130
RECOVERY MANAGER
132
OPERATING SYSTEM
120
SWITCH
121
(LOCAL CONNECTION)
(MODEM CONNECTION)
123
128
MODEM
127
126
MODEM
129
124
REMOTE CLIENT
130
RECOVERY MANAGER
132
OPERATING SYSTEM

FIG. 1

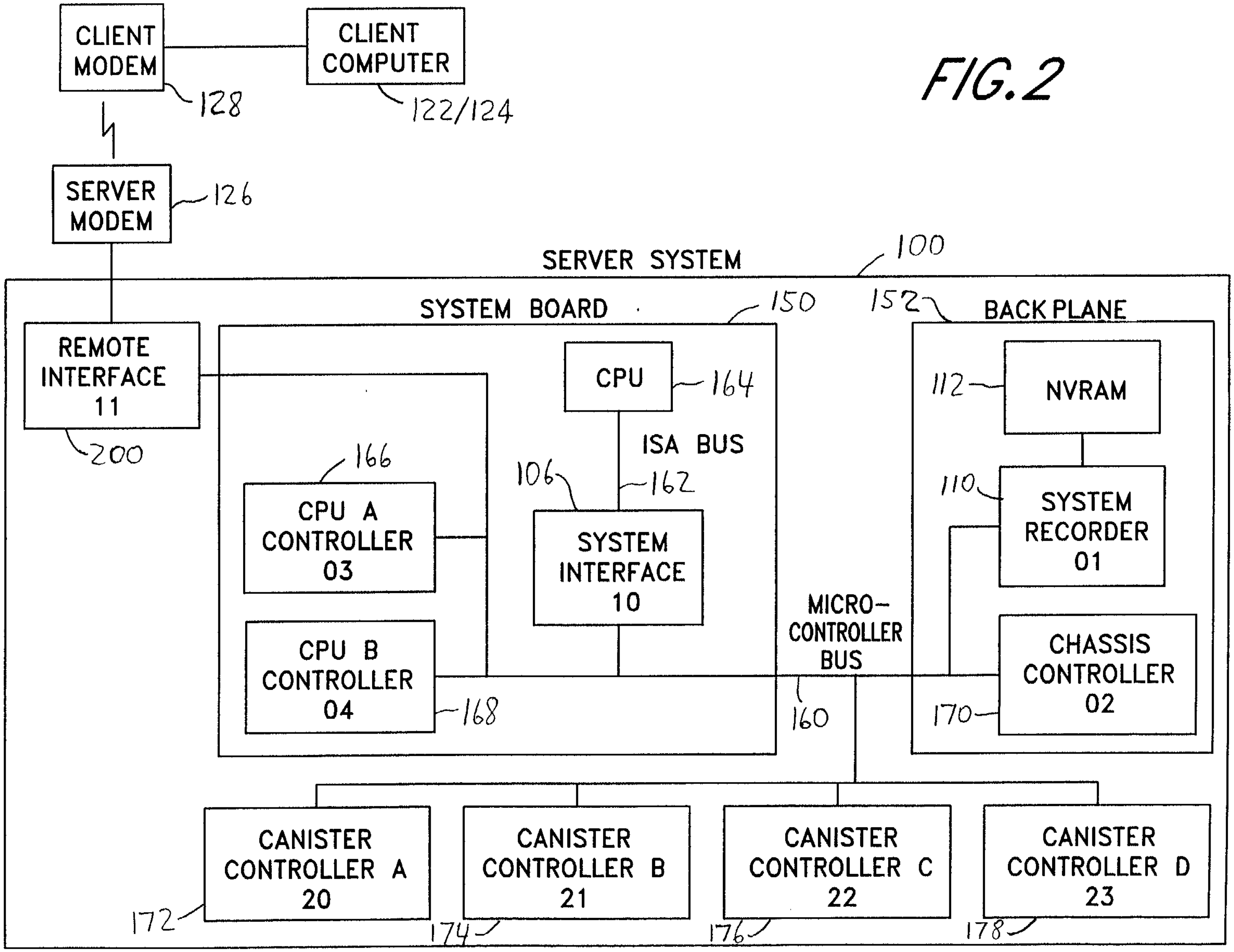

CLIENT MODEM
128
CLIENT COMPUTER
122/124
SERVER MODEM
126
SERVER SYSTEM
100
SYSTEM BOARD
150
152
BACK PLANE
REMOTE INTERFACE 11
200
CPU
164
ISA BUS
106
162
CPU A CONTROLLER 03
166
SYSTEM INTERFACE 10
CPU B CONTROLLER 04
168
MICRO-CONTROLLER BUS
160
112
NVRAM
110
SYSTEM RECORDER 01
CHASSIS CONTROLLER 02
170
CANISTER CONTROLLER A 20
172
CANISTER CONTROLLER B 21
174
CANISTER CONTROLLER C 22
176
CANISTER CONTROLLER D 23
178

FIG. 2

REMOTE INTERFACE BOARD AND CONNECTIONS

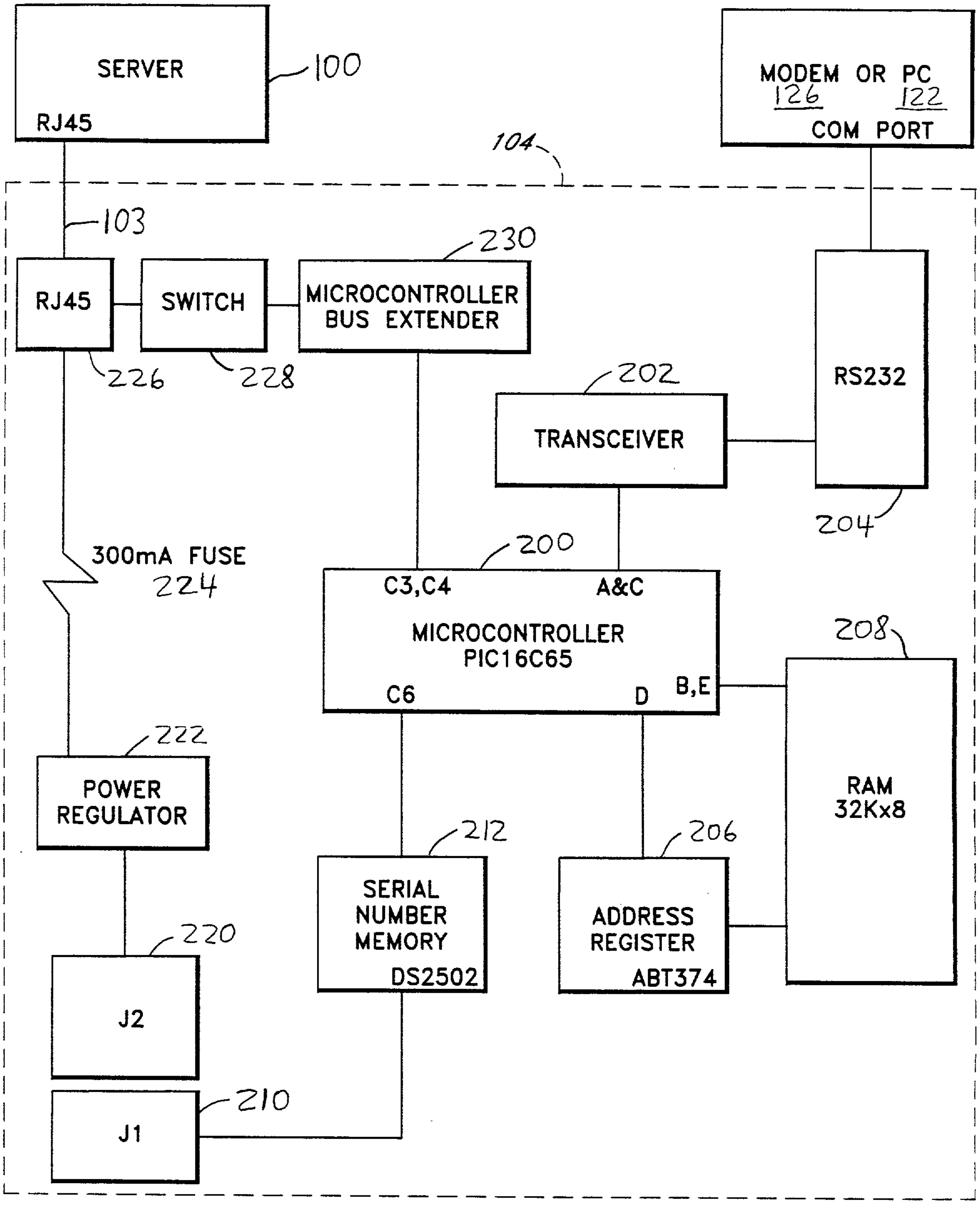


FIG. 3

Remote Interface Serial Protocol Message Formats

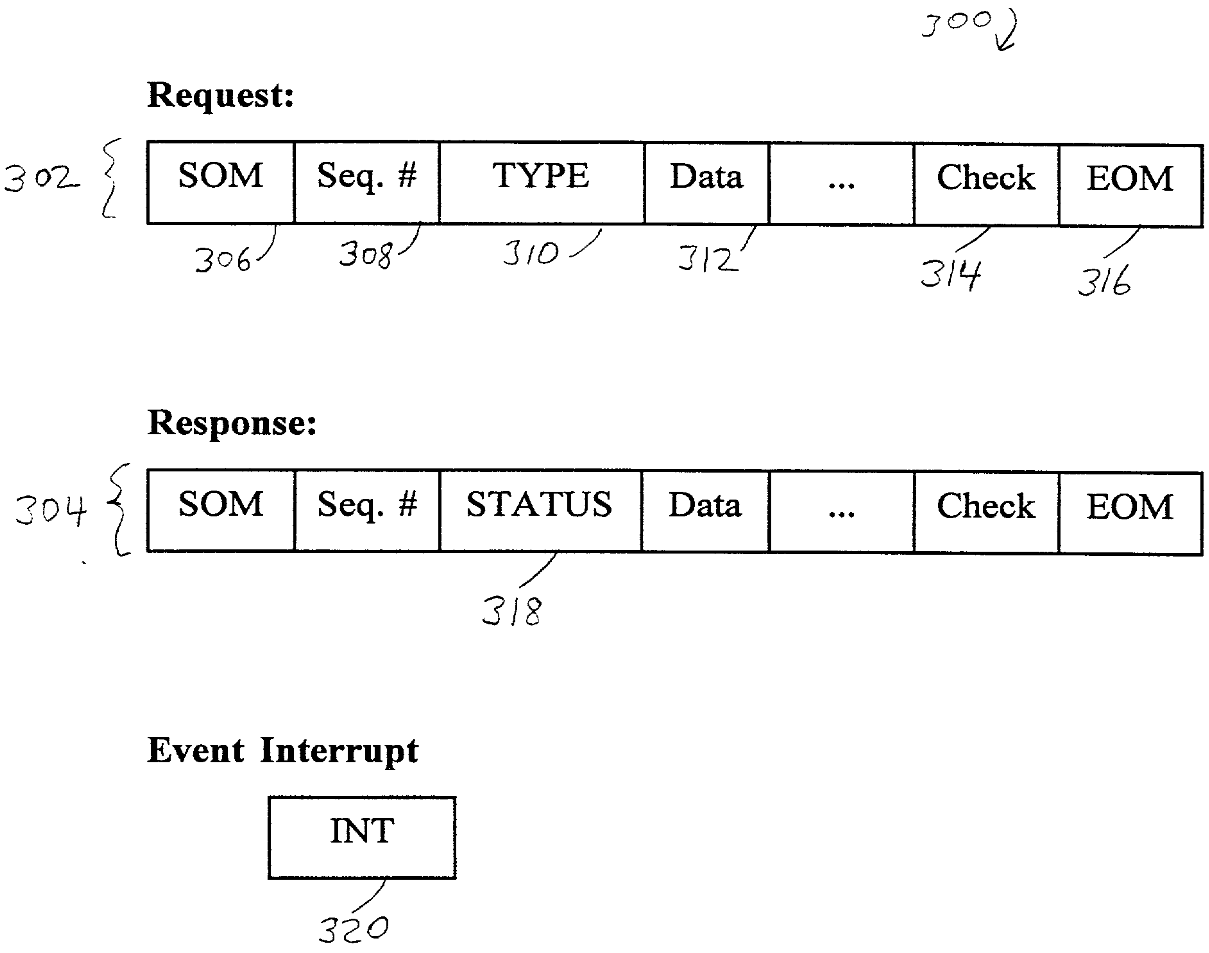


FIG. 4

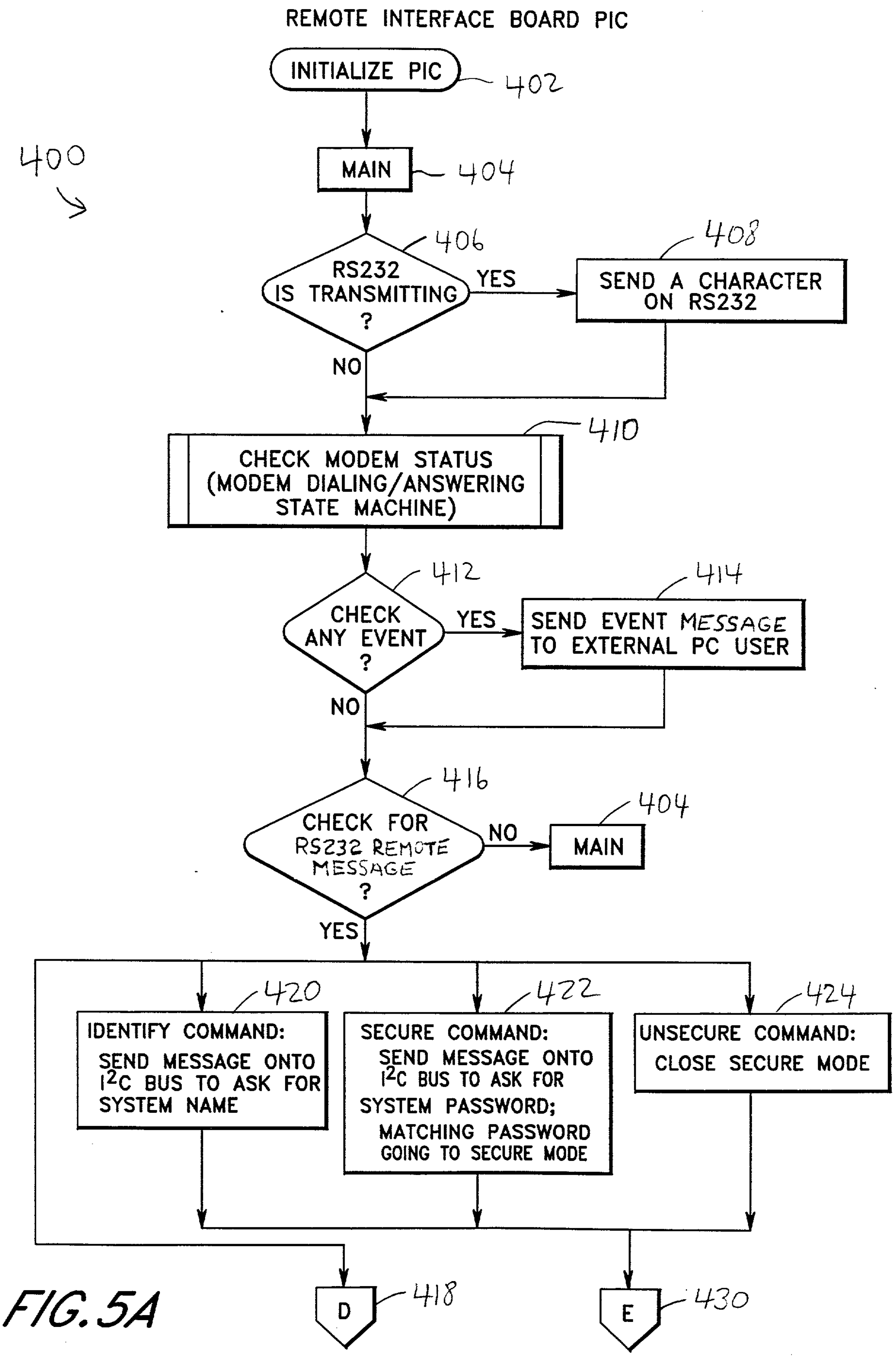

REMOTE INTERFACE BOARD PIC
INITIALIZE PIC
402
400
MAIN
404
406
RS232 IS TRANSMITTING ?
YES
408
SEND A CHARACTER ON RS232
NO
410
CHECK MODEM STATUS (MODEM DIALING/ANSWERING STATE MACHINE)
412
CHECK ANY EVENT ?
YES
414
SEND EVENT MESSAGE TO EXTERNAL PC USER
NO
416
CHECK FOR RS232 REMOTE MESSAGE ?
NO
404
MAIN
YES
420
IDENTIFY COMMAND: SEND MESSAGE ONTO I2C BUS TO ASK FOR SYSTEM NAME
422
SECURE COMMAND: SEND MESSAGE ONTO I2C BUS TO ASK FOR SYSTEM PASSWORD; MATCHING PASSWORD GOING TO SECURE MODE
424
UNSECURE COMMAND: CLOSE SECURE MODE
D
418
E
430

FIG. 5A

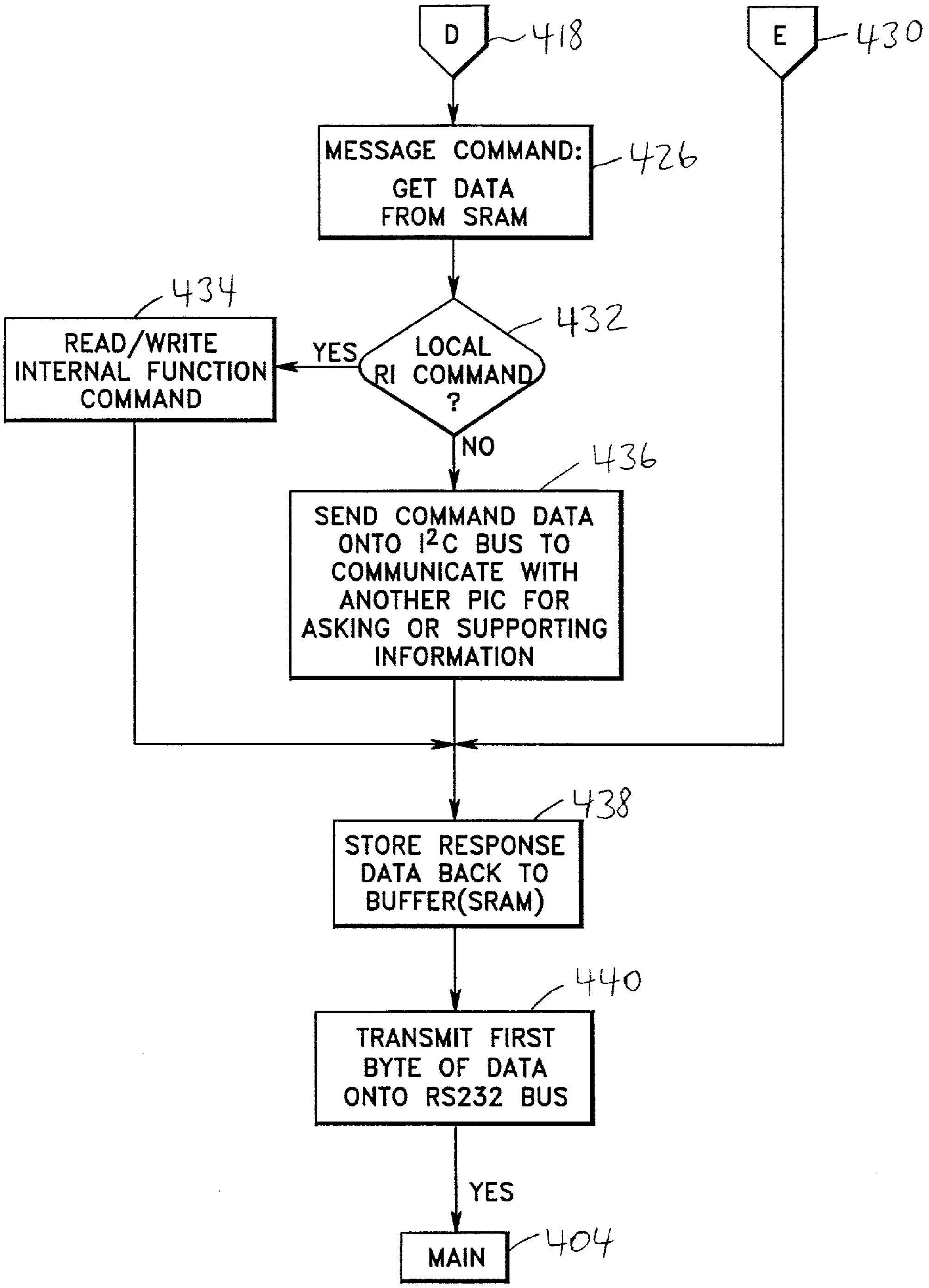
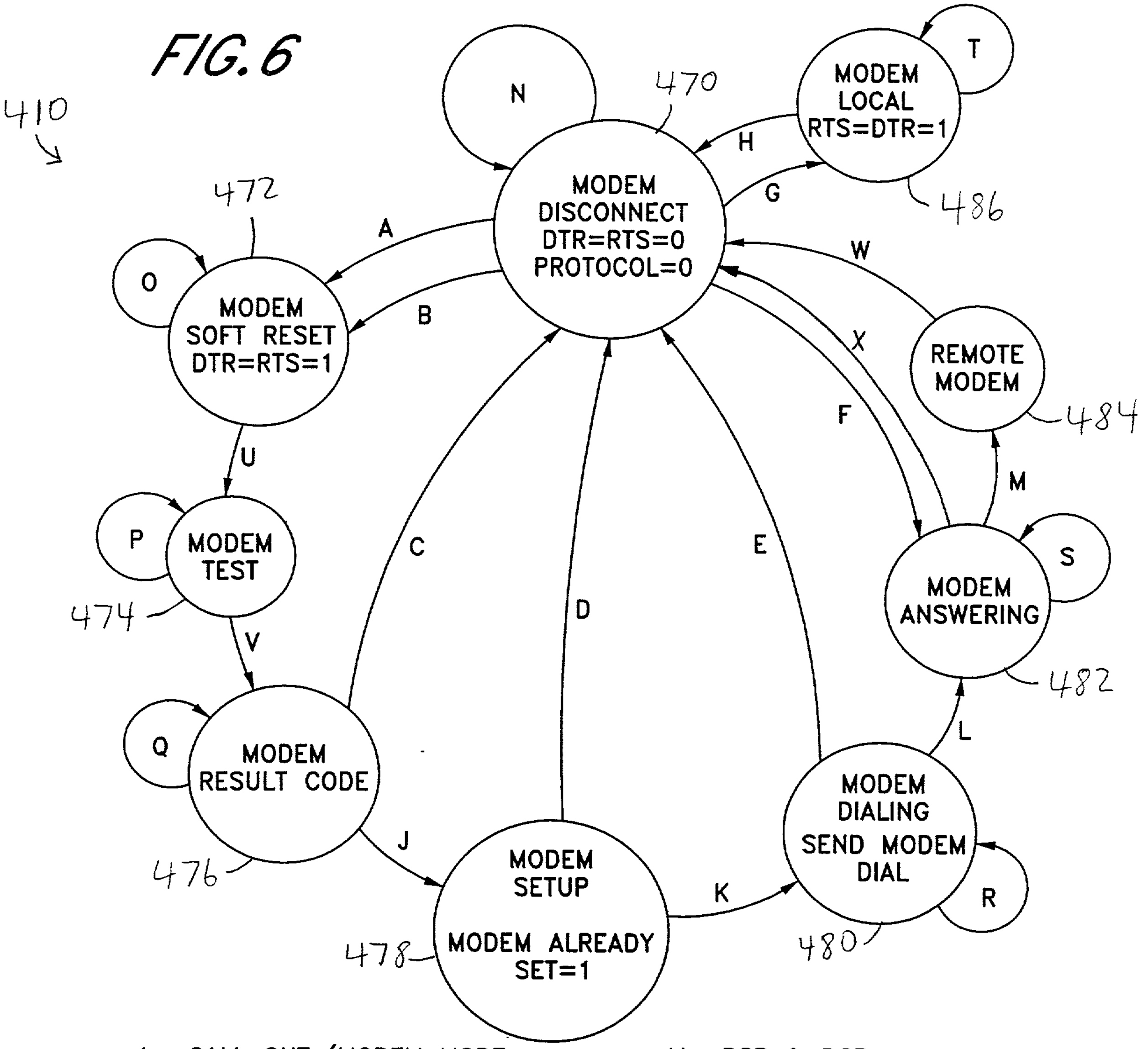

D
418
E
430
MESSAGE COMMAND:
GET DATA
FROM SRAM
426
434
READ/WRITE
INTERNAL FUNCTION
COMMAND
YES
LOCAL
RI COMMAND
?
432
NO
436
SEND COMMAND DATA
ONTO I2C BUS TO
COMMUNICATE WITH
ANOTHER PIC FOR
ASKING OR SUPPORTING
INFORMATION
438
STORE RESPONSE
DATA BACK TO
BUFFER(SRAM)
440
TRANSMIT FIRST
BYTE OF DATA
ONTO RS232 BUS
YES
MAIN
404

FIG.5B

MODEM DIALING AND ANSWERING STATE MACHINE

FIG. 6



A– CALL OUT (MODEM MODE
& MODEM ALREADY SET
& CTS & nDIAL RETRIES)
B– SETUP (MODEM MODE
&!MODEM ALREADY SET & CTS)
C– !RESULT STATUS OK
D– !nDIAL RETRIES
E– !DIAL OK
F– RINGING (MODEM MODE
& MODEM ALREADY SET
& CTS & RING)
G– DSR
H– (LOCAL & MODEM MODE) V !DSR
J– RESULT STATUS OK
K– nDIAL RETRIES
L– DIAL OK

M– DSR & DCD
N– !CTS
V !nDIAL RETRIES
V !RING
V !DSR
O– !SEND STRING DONE
P– !SEND STRING DONE
Q– !MODEM RESULT
STATUS DONE
R– !PREV EOS & !2 SECONDS
S– !DSR
V !DCD
T– DSR
U– SEND STRING DONE
V– SEND STRING DONE
W– !DCD
X– !DCD & TIMEOUT

# APPENDIX B

Provisional Patent Application

6391-710:

Title: REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT SYSTEM

Invs: Karl Johnson
Tahir Sheik

The following documents are attached and form part of this disclosure:

1. *Maestro Recovery Manager Analysis - Problem Statement*, pp. 1-10.

2. *Remote Interface Board Specification,* Revision 2 13-000072-01, June 21, 1996, pp. 1-11.

Multiple Node Service Processor Network

A means is provided by which individual components of a system are monitored and controlled through a set of independent, programmable microcontrollers interconnected through a network. Further means are provided to allow access to the microcontrollers and the interconnecting network by software running on the host processor.

Fly-by-wire

A means is provided by which all indicators, push buttons and other physical control means are actuated via the multiple node service processor network. No indicators, push buttons or other physical control means are physically connected to the device which they control, but are connected to a microcontroller, which then actuates the control or provides the information being monitored.

Self-Managing Intelligence

A means is provided by which devices are managed by the microcontrollers in a multiple node service processor network by software running on one or more microcontrollers, communicating via the interconnecting network. Management of these devices is done entirely by the service processor network, without action or intervention by system software or an external agent.

Flight Recorder

A means is provided for recording system events in a non-volatile memory, which may be examined by external agents. Such memory may be examined by agents external to the network interconnecting the microcontrollers.

Replicated components: no single point of failure

A means is provided by which no single component failure renders the monitoring and control capability of the system inoperable.

Extension by serial or modem gateway

A means is provided allowing an external agent to communicate with the microcontrollers by extending the interconnecting network beyond the physical system.

The following provisional patent applications, commonly owned and filed on the same day as the present application, are related to the present application and are incorporated by reference:

COMPUTER SYSTEM HARDWARE INFRASTRUCTURE FOR HOT PLUGGING MULTI-FUNCTION PCI CARDS WITH EMBEDDED BRIDGES (6391-704); invented by:

Don Agneta
Stephen E.J. Papa
Michael Henderson
Dennis H. Smith
Carlton G. Amdahl
Walter A. Wallach

COMPUTER SYSTEM HARDWARE INFRASTRUCTURE FOR HOT PLUGGING SINGLE AND MULTI-FUNCTION PC CARDS WITHOUT EMBEDDED BRIDGES (6391-705); invented by:

Don Agneta
Stephen E.J. Papa
Michael Henderson
Dennis H. Smith
Carlton G. Amdahl
Walter A. Wallach

ISOLATED INTERRUPT STRUCTURE FOR INPUT/OUTPUT ARCHITECTURE (6391-706); invented by:

Dennis H. Smith
Stephen E.J. Papa

THREE BUS SERVER ARCHITECTURE WITH A LEGACY PCI BUS AND MIRRORED I/O PCI BUSES (6391-707); invented by:

Dennis H. Smith
Carlton G. Amdahl
Don Agneta

HOT PLUG SOFTWARE ARCHITECTURE FOR OFF THE SHELF OPERATING SYSTEMS (6391-708); invented by:

Walter A. Wallach
Mehrdad Khalili
Mallikarunan Mahalingam
John Reed

REMOTE SOFTWARE FOR MONITORING AND MANAGING ENVIRONMENTAL MANAGEMENT SYSTEM (6391-709); invented by:

Ahmad Nouri

REMOTE ACCESS AND CONTROL OF ENVIRONMENTAL MANAGEMENT SYSTEM (6391-710); invented by:

Karl Johnson
Tahir Sheik

HIGH PERFORMANCE NETWORK SERVER SYSTEM MANAGEMENT INTERFACE (6391-711); invented by:

Srikumar Chari
Kenneth Bright
Bruno Sartirana

CLUSTERING OF COMPUTER SYSTEMS USING UNIFORM OBJECT NAMING AND DISTRIBUTED SOFTWARE FOR LOCATING OBJECTS (6391-712); invented by:

Walter A. Wallach
Bruce Findley

MEANS FOR ALLOWING TWO OR MORE NETWORK INTERFACE CONTROLLER CARDS TO APPEAR AS ONE CARD TO AN OPERATING SYSTEM (6391-713); invented by:

Walter A. Wallach
Mallikarunan Mahalingam

HARWARE AND SOFTWARE ARCHITECTURE FOR INTER-CONNECTING AN ENVIRONMENTAL MANAGEMENT SYSTEM WITH A REMOTE INTERFACE (6391-714); invented by:

Karl Johnson
Walter A. Wallach
Dennis H. Smith
Carl G. Amdahl

SELF MANAGEMENT PROTOCOL FOR A FLY-BY-WIRE SERVICE PROCESSOR (6391-715); invented by:

Karl Johnson
Walter A. Wallach
Dennis H. Smith
Carl G. Amdahl

# Problem Statement

## ♦ Introduction

Maestro Recovery Manager(MRM) is a software which locally or remotely manage a Raptor when a server is down or up, operating system died, LAN communication failed, or other server components failed .

User will be able to manage the server in very simple, usable, and friendly GUI environment. MRM use modem for remote and serial communication port for local to communicate with server for diagnostic and recovery.

Primary role of remote management is diagnosing and restoring service as quickly as possible in case of a service failure.

System administrator, LAN administrator in customer shop and NetFrame Technical support will be primary user for the system.

## ♦ Requirement Sources

MRM requirements comes from the following

1 - Focus Group (Customer Support and Training )
2 - User Walkthrough held by MRM team and Customer Support in Dec 96
3 - Down System Management Road map (96)
This road map is preliminary road map combined with Up System Management road map.
4 - MRM Road Map 97-98
This Road Map presented to Engineering Council Meeting on Mar 10, 1997.

5 - Raptor System, A Bird's Eye View.
6 - Raptor Wire Service Architecture

The following requirements have been identified for MRM

## ♦ Support Remote Management for Diagnostic and Recovery

Remote Management cover remote access to the Raptor Out Of Band management features. Remote Management will use Out of Band ,**Control Diagnostic and Monitor Subsystem (CDM)** remote management to cover the other high value added remote management functions. primary role of remote management is diagnosing and restoring service as quickly as possible in case of service failure.

## ♦ Support Remote Management ... (continue)

The control of Raptor is completely “Fly By Wire” - i.e. no physical switch directly controls any function and no indicator is directly controlled by system hardware. All such functions referred to as “Out of Band “ functions are controlled through a CDM. CDM basic functions are available so long as A/C power is available at the input to any of the power supplies.

CDM Subsystem supervises or monitors the following system features.

- **Power supplies** - Presence, status, A/C good, Power on/off and output voltage.
- **Environment** - Ambient and exhaust temperatures, Fan speed, speed control, Fan fault and overtemp indicators.
- **Processor** - CPU Presence, Power OK, Overtemp and Fault, NMI control, System reset, Memory type/ location and Bus/Core speed ratio.
- **I/O** - I/O canister insertion/removal and status indicator , PCI card presence, PCI card power and smart I/O processor Out Of Band control.
- **Historical** - Log of all events, Character mode screen image, and Serial number

## ♦ Support for Object Oriented Graphic User Interface

OO-GUI is graphic user interface with the following characteristic.

- **User task oriented**
  It uses tasks which user familiar and daily working with. User does not need to learn the tasks.
- **User objects**
  It uses objects which user working with during her or his daily work.
- **Simplicity and useablity**
  It is very simple to use and does not need long learning period.
- **Point and click with context sensitive help**
  Context sensitive help and point and click will help user to be very productive and get any information he needs on specific object or field or subject.
- **Drag and drop**
  Drag and Drop capability works with user object very well to accomplish the tasks.

## ♦ Release Requirements (MRM V2.0, 4Q96)

**Maestro Recovery Manager** (MRM )will support the following features **locally** through serial port and Wire Service Remote Interface card on the **Raptor16.**

MRM provide user friendly GUI with point and click capability to perform the following tasks which reviewed and accepted by the **Focus Group** for 4Q96 release.

- **Power On /Off**
  MRM support Power On/Off the server.
  User can do this task by right mouse click on the server object in the screen and see the result.

- **Display Flight Recorder.**
  While the server is working , Wire Service record all the server information in the 64K NVRAM. After the server failed, MRM will display the system log recorded in the NVRAM. User can evaluate the information and find the cause for the server failure. This can be done by right mouse click on the Flight Recorder object in the screen.

- **System Reset**

  MRM support rebooting the server by right mouse click on the server object in the screen. This is warm reboot of the server and works as pushing the "reset" button on the server.

- **Save**

  MRM will support saving Flight Recorder data, so user can send the file to the technical support for further diagnostic and recovery. It also can save the response for any Wire Service command failure.

- **On Line help**
  MRM will support online help contains overview, Getting Started, MRM tasks, Diagnostic and Recovery, and BIOS help.

- **B0 back plane support**

  MRM will support the server with B0 back plane . Server with B0 back plane display wrong time stamp. MRM uses NetWare 4.11 Operating system time stamp to display correct time stamp.

## ♦ Release Requirements (MRM V2.1, 1Q97)

**Maestro Recovery Manager (MRM) will support Raptor16 Phase 2** for next release as follow. This release will delivered to customer by NetFrame Customer Support on CD.

### MRM V2.1

**MRM V 2.1** will support the **MRM V2.0** **plus** the following **new** features for next release.

- **User Walkthrough Requirements held on Dec 17, 1996**
- **Recovery and Diagnostic help.**
  This help enable the user to display help based on message source or severity (fatal error, error, warning, ). In each case the help inform the user the cause for the error and what steps to take to solve the problem.
- **C0 /E18 back plane support**
- **New C0 back plane Wire Service, Diagnostic, and BIOS message structure**

## ♦ Release Requirements (MRM V2.2 , 2Q97)

### MRM V2.2 for Raptor 16

MRM V2.2 will support MRM V2.1 plus the following new features.

- **Remote connection via modem**

  MRM supports remote connection to an NF9000-16 via an external modem. MRM needs one external modem for client side and one external modem for the server side. The client modem can be installed and set up via the Windows NT/95 standard control panel/Modems installation. The server side modem has to be set up and connected to the server. Details of installation and setup for the modem are provided in the **NF9000 Maestro Recovery Manager Installation Guide.**
  **MRM does not support internal modems.**

  The following **external Hayes compatible** modems have been tested and worked with MRM.

  * **Client Modem**
  US Robotics Sportster 33.6 Fax modem
  ZOOM fax MODEM V.34X 33.6

  * **Server Modem**
  ZOOM fax MODEM V.34X 33.6

- **System Status**

  MRM supports **retrieve and update** of the system status components.
  System status comprised of the following components.

  * **Power Supplies**

  The following information will be displayed for this feature.

  1. Presence
  2. Status(ACOK, DCOK)
  3. Power On/off
  4. Output voltage (Analog measure of main supply + VREF)

## * Temperatures

We will support four types of temperature for 5 sensors and display Operating (10 -35 degree C) and None-operating (-40 to 70 degree C).

1. Temperature of all sensors
2. Warning temperature
3. Shutdown temperature
4. System over temp

## * Fans

There are different type of fans in the system such as system fan and canister fan. All of them have the common following characteristics.

5. Speed (speed data)
6. Control (LOLIM, can be set to LOW or HIGH)
7. Fault ( LED, Bits )

## * Processors

There are 4 CPU in the Raptor16 with the following parameters.

1. CPU presence
2. CPU Power OK
3. System over temp
4. System Fault
   If system over temp or CPU internal error or system power failure.
   then wire service report System Fault
5. CPU Error
   If internal CPU error occurred , then report CPU error
6. CPU NMI control
7. System Board Bus/Core speed ratio

## * I/O Canisters

There are four canisters available

1. I/O canister (insertion, removal)
   This shows presence bits for canister.
2. PCI cards
   This reflect PCI card slots [1-4] presence
3. PCI card power
   This controls canister PCI slot power

- **Serial Numbers**

  This is the last known serial data for the following server parts

  1. Back plane
  2. Canister 1-4
  3. Remote Interface (not implemented)
  4. System Board
  5. Power supply 1-2

- **Revisions**

  MRM will support the following chips revision

  1. Back Plane
  2. System board
  3. Power Supply 1- 2
  4. Canisters 1- 4
  5. Local Interface
  6. Remote Interface

## • Context-sensitive Help

All elements in the window such as icon, entry field, push button, and radio button have context-sensitive help. This help contains the following type.

- **What's this**

  It shows description of each elements in the window which it is not disabled. This can be accomplished by right mouse click on each element in the window.

- **Help push button.**

  This display general help for all windows.

- **F1 Key**

  The key displays the help for any entry field in the window.

## • Print

MRM supports printing of flight recorder based on all messages, warning & errors, and errors with one type of font.

- **Password**

  Wire Service password is originally set by Manufacturing to "NETFRAME" (case sensitive) for every NF9000-16 server.
  MRM provides a **password changing** mechanism for the Wire Service system.
  For security purposes , MRM only allows the password to be changed via the **local** serial port connection and **not** via the **remote** connection

- **Support B0/E18 on NT4.0 server**

  MRM supports **B0/E18** configurations by utilizing a time stamp software component which resides on the NT4.0 server.
  Installation instructions for the time stamp are provided in the **NTReadMe** file on a floppy disk packaged with MRM.
  MRM requires the NetFRAME NT Value Add software to operate.
  The NetFRAME NT Value Add software will automatically install the time stamp for you. If you have not installed NetFRAME NT Value Add, then you need to install the time stamp provided for you on the NTSup floppy disk.

- **Support for InstallShield**

  InstallShield setup software is used to install MRM on the client workstation.

- **Delivery**

MRM package contains the following.

* **NF9000 Maestro Recovery Manager CD release.**
  This CD contains MRM software and documentation.
* **Two support floppy disks for NF9000-16 B0 back plane for NT and NetWare.**
* **Boxes contain above items , Remote Interface Card, adapter, cables, and documentation.**

- **Dependency**

**MRM version 2.2** depends on the following items:

- Remote Interface chip provided by Wire Service(Firm Ware) department.
- Remote Interface card provided by Hardware Engineering department.
- Remote Interface boxes, cables, and power adapters provided by Manufacturing.

## ♦ Release Requirements (MRM V2.2, 2Q97)

### MRM V2.2 for Raptor 8

MRM V2.2 for Raptor 8 has the same features as MRM v2.2 for Raptor16 with the following different.

- **Support for C0 back plane and F18 BIOS**
- **System Status**

  The following components of System Status are different from MRM V2.2 for Raptor16.

  - **Power Supplies**

    1. User can not turn off and on specific power supply.
    2. Raptor 8 has three power supply.
    3. There are no DC (OK, BAD) for Raptor8.
    4. AC for all power supplies are good all the times.

  - **Fans**

    1. Four system board fans in front
    2. Two system board fans (Storage fans) in back
    3. Group A and group B sharing two fans.

  - **I/O Groups**

    1. Group A contains 4 PCI card slots
    2. Group B contains 4 PCI card slots.

  - **Serial Numbers**

    1. Serial number for Group A and B fans are the same.
    2. There is serial number for power supply # 3.

  - **Revisions**

    1. Group A and B fans have the same revision.
    2. There is revision for power supply #3

- **Delivery**

MRM package contains the following.

- **NF9000 Maestro Recovery Manager CD release.**
  This CD contains MRM software and documentation.

- **Boxes contain above items , Remote Interface Card, adapter, cables, and documentation.**

# Remote

# Interface Board

# Specification

Revision 2
13-000072-01
June 21, 1996

Tahir Sheikh
NetFRAME Systems, Inc.

# Index

## Overview:

This board is an interface between Raptor Wire Services and an external modem. The system status and commands are passed through the RS232 connection at the modem side to the Wire Services bus, the I2C bus, controlled through an on board PIC16C65. The I2C signals are translated by the PIC16C65 into an eight signal RS232 protocol and passed through a voltage level translator LT1133A, with baud capable of reaching the speed of 120k. A 25 pin D-Sub connector resides on the other side of the voltage level translator.

The system status storage is through a 32Kx8 SRAM, with an external lath for latching the higher addressing bits of the data RAM. A signal powered EPROM is used for storing board ID information.

The board is powered through 7.5V and 700mA supply unit, and is an alternative source for the bias powered partition of the Wire Services. The bias powered block includes an NV-RAM and a PIC16C65 which are resident on the Raptor back plane. The power source is  regulated through a high frequency switching regulator.

## 1.0 Features

The designed features are as follows:

### 1.1 I2C Interface

The two wires interface is brought from the Raptor and passed to the PIC16C65 using an RJ45. A bus extender 82B715 is connected between the external interface to the local I2C bus. Port C bit 3 is  the clocking bit, and Port C bit 4 is the data line.

### 1.2 RS232 Protocol

The communication with the modem is based on the RS232. Microcontroller PIC16C65 is used to generate the receive and the transmit signals, where the signal levels are transposed to the RS232 levels by the LT1133A. The 3 transmit signals, RTS, SOUT and DTR are from Port A bits 2, 3 and 4, where as the 5 receive signals are from two ports, DCD, DSR from Port C 1,0 and SIN, CTS and RI from Port A 5, 0, 1.

The 25 pin RS232 pin connection is used instead a 9 pin connector, since this type of connector is more common than the other. All the extra pins are no connect except the pins 1 and 7, where pin 1 is chassis ground and pin 7 is a signal ground.

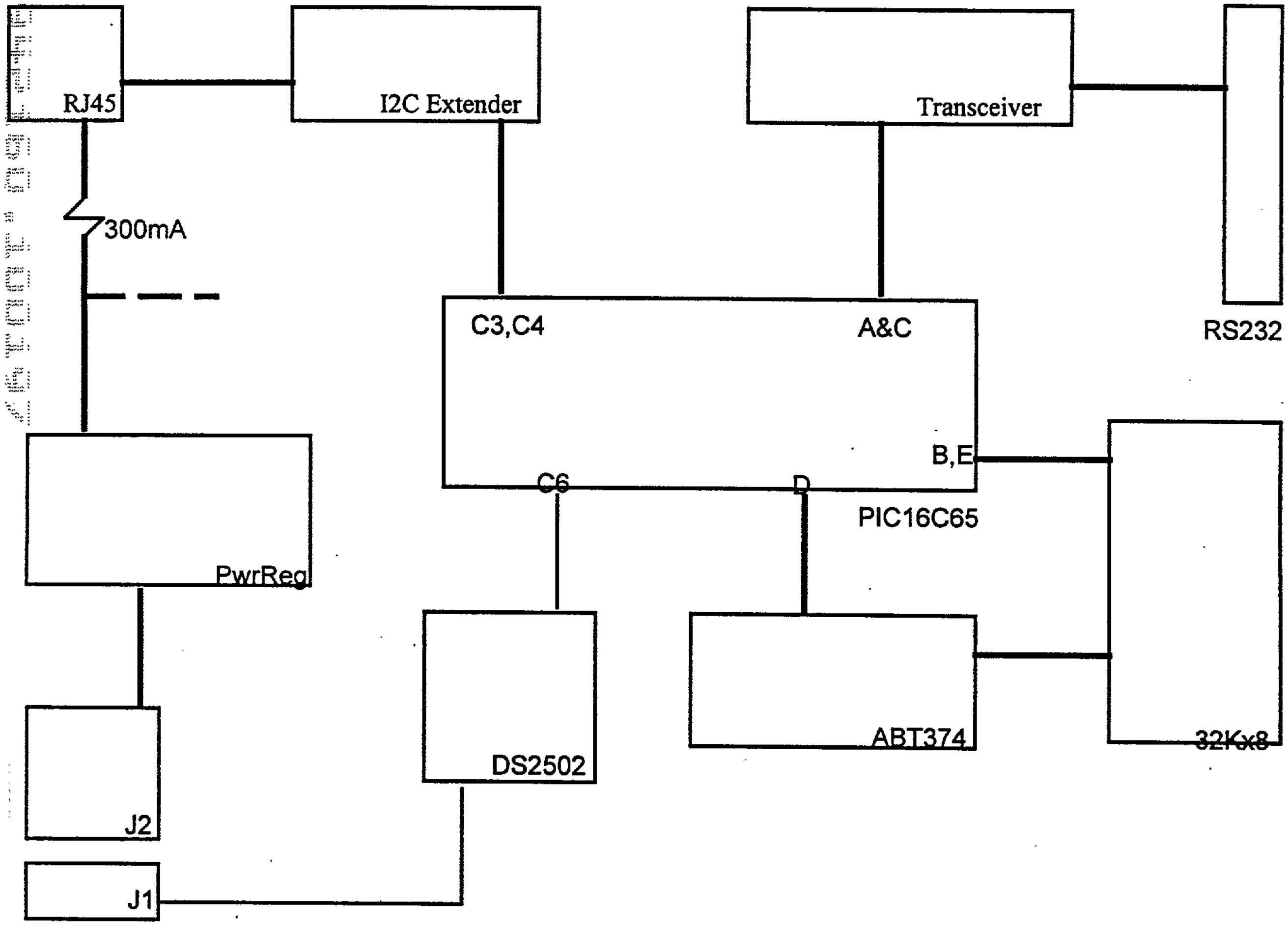
The connection through LT1133A can be run up to 120k Baud and is ESD protected to +/- 10kV.

The short voltage at the output can be +/- 30V and is isolated to the forward direction only.

### 1.3 PIC16C65 and 32Kx8

A 32Kx8 SRAM is available for storage and transfer between the internal Wire Services and the external remote interface. Port D is the address port, while an external 74ABT374 is for expanding the address range to 15 bits. Port B is the data bus for the bi-directional data interconnect. Port E is for the SRAM enable, output tristate and the write control signals.

The PIC16C65 is designed for a frequency of 12MHz. An LED is also connected to the Port C bit 5.



## Figure 1: Remote Interface Interconnect

### 1.4 Serial ID EPROM

DS2502 is for storing board ID, connected to PIC16C65 Port C bit 6. The programming is handled through a jumper applied through connector J1. DS2502 is a signal powered, retaining the charge into a capacitor, sourced through the data line.

### 2.1 Alternative Power Source

The board is powered through 7.5V and 700mA (or 800mA which ever available) supply unit. After regulating the supply, it is an alternative source for the bias powered partition of the Raptor Wire Services. The bias powered block includes an NV-RAM and a PIC16C65 which are resident on the Raptor back plane.

The power source is regulated through a high frequency switching regulator based on Linear Technology LT1376. The input to the regulator circuitry is off a wall mounted adapter. The regulated output is consumed locally and 300mA are sourced to the Raptor Wire Services through a fuse and an RJ45 P1.

### 2.2 Power Consumption

The following is an average estimated power consumption with the board running at a base frequency of 12MHz.

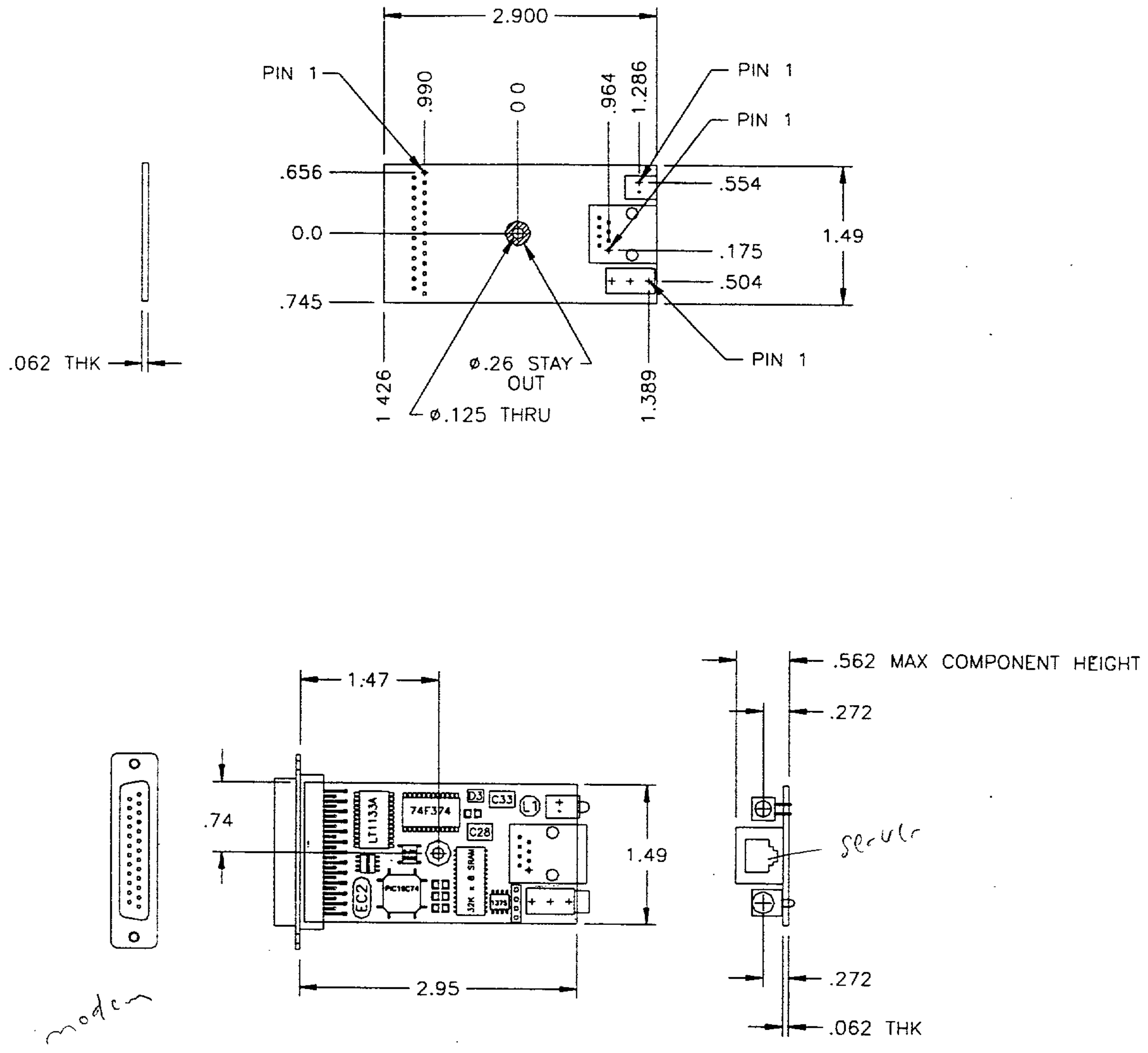
| | | |
|---|---|---|
| PIC16C65 | Microcontroller | 30mA |
| 82B715 | I2C Extender | 10mA |
| 32Kx8 | SRAM | 80mA |
| LT1133A | Transceiver | 70mA |
| 374 | Latch | 30mA |
| | Misc. | 60mA |
| | | --------------- |
| | | 280mA |
| | Alt. Source | 300mA |

----------------

580mA

LT1376, L1, D2, D3, etc. 1.45 Watts max.

==============================



## Figure 2: Mechanical Orientation

## 2.3 Board Layout

The board is based on controlled impedance of 60 Ohms +/- 10%, with 6 layers and test points for all signals. The width is restricted by the dimension of the RS232 due to the mounting constrains. The board is dual sided with active components kept on the top side only.

The high frequency bypass is kept with .1uf and .001uf, where the charge storage is kept by two 33uf and two 1uf capacitors.

The location and mounting of the power connector and the LED are kept such that the both sides of the cabinet are identical, therefore interchangeable.

## 3: Enclosure

The enclosure is planned to be Injection Molded Aluminum, a side view is in figure 3. Aluminum instead of plastic is selected due to the regulator heat, and EMI shielding.

The board connects at three locations between the top and the bottom enclosures. Two locations are based on the clamp shell design at the D-Sub and the RJ45, two opposite ends of the enclosure. The third location is a mounting hole in the center of the enclosure.

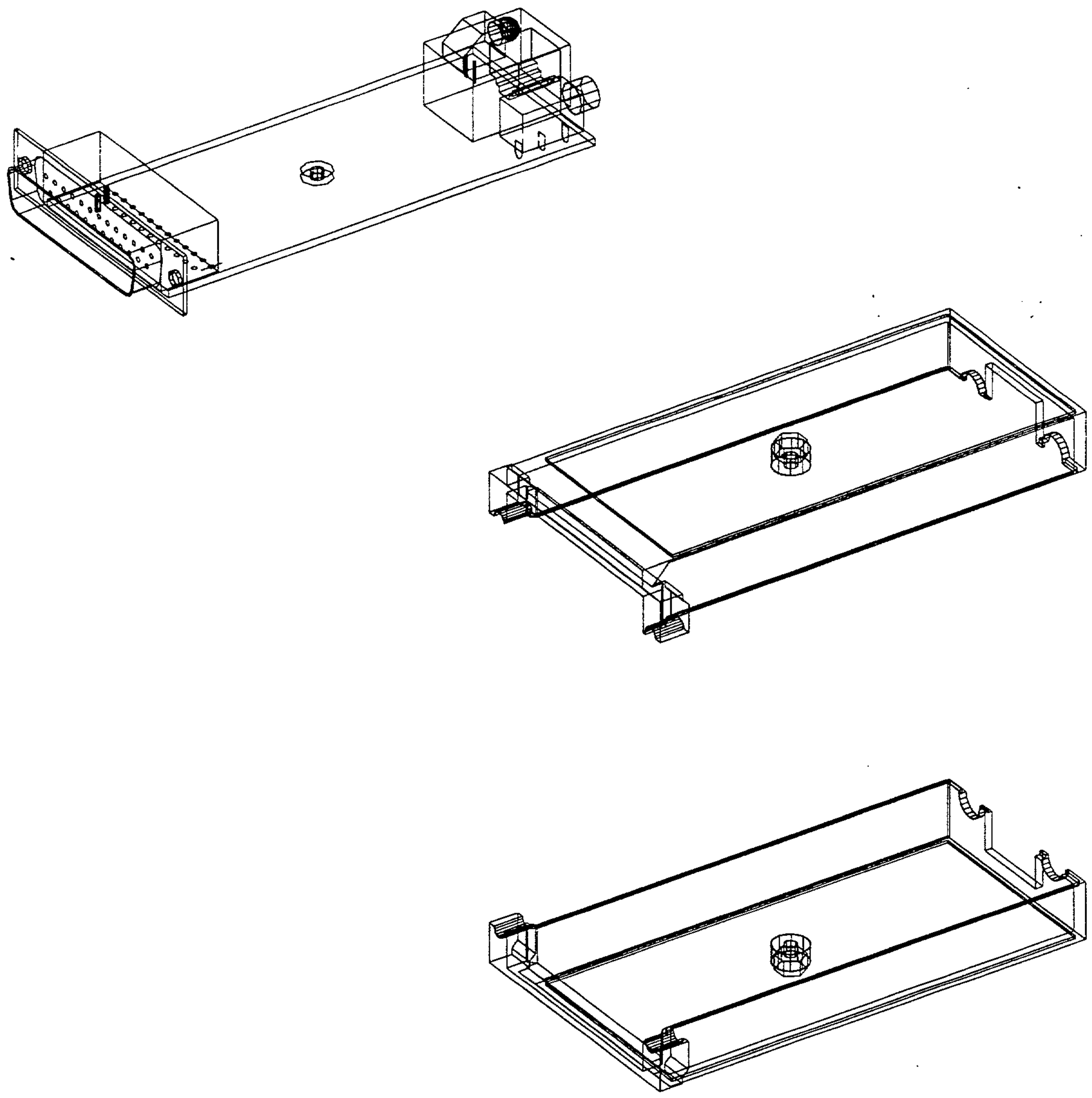


Figure 3: Board and Enclosure Isometric View

## 4.0 Environment

The environmental specification is based on assumptions:

The environment is Ground Fixed.
The "Quality Level of II" is used.
Bellcore Method I, Parts Count Method, Case 2 for prediction.
Burn-in time 120 hours.
Operated at 40C and 50% rated electrical stress.

## 4.1 Environmental Specification:

### MEAN TIME BETWEEN FAILURE

*2.445250e+06 Hours*

This number is calculated based on the Bellcore Technical Reference TR-NWT-000332, Reliability Prediction Procedure for Electronic Equipment, Issue 4, September 1992.

### ALTITUDE

| | |
|---|---|
| Operating | -100 to 10,000 feet |
| Non-Operating | -100 to 40,000 feet |

### HUMIDITY

| | |
|---|---|
| Operating | 10% to 80% R.H., Maximum Gradient 10% per hour |
| Non-Operating | 5% to 90% R.H., Maximum Gradient 10% per hour |

### TEMPERATURE (ambient)

| | |
|---|---|
| Operating | 10 to 40 degrees C Maximum Gradient 10 degrees C per hour |
| Non-Operating | -40 to 70 degrees C Maximum Gradient 10 degrees C per hour |

### SHOCK

Operating

| | |
|---|---|
| Magnitude | 2 G's (peak) |
| Duration 11 ms | |
| Waveform | Half Sine |

Non-Operating

| | |
|---|---|
| Magnitude | 10 G's (peak) |
| Duration 11 ms | |
| Waveform | Half Sine |

### VIBRATION

Operating

| | |
|---|---|
| Frequency Range | 5 to 500 Hz |
| Magnitude | 0.010 inch peak to peak displacement |
| Acceleration | 0.20 GÕs peak |

Non-Operating

| | |
|---|---|
| Frequency Range | 5 to 500 Hz |
| Magnitude | 0.010 inch peak to peak displacement |
| Acceleration | 0.50 GÕs peak |

## DROP (PACKAGED)

ASTM D4169

## ELECTRICAL

| | |
|---|---|
| Nominal Line | 115 VAC or 230 VAC @ 50/60 Hz autoranging |
| Line Deviation | 90-130 VAC & 180-256 VAC @ 47-63 Hz |
| Line Transient/Surge Susceptibility | 1.25 x highest rated nominal voltage or 300 Vrms, whichever is less, for 1 second. |

## ELECTRO-MAGNETIC COMPATIBILITY

FCC, Class A under FCC Rule 15, Subpart B, conducted and radiated.

Canadian Radio Interference Regulations, C.R.C., c.1374, Sec. 2, as amended in The Canadian Gazette, Part II, Vol. 122, No. 20, dated Sept. 28, 1988.

European EMC Directive (89/336/EEC) CISPR 22 (Class B).

IEC 801-2:1984 8 kV air discharge

IEC 801-3:1984 3 V/m, 27-500 MHz

IEC 801-4:1988 1 kV mains, 500 V other.

## SAFETY AGENCIES

UL, CSA, VDE, JIS

## Electrostatic Discharge

| | | |
|---|---|---|
| Air Discharge | 2.5 to 5.0KV | no errors allowed |
| | 5.1 to 10.0 KV | recoverable errors through system allowed |
| | 10.0 to 20.0KV | recoverable errors through power cycling allowed |
| Contact Discharge | 0 to 8.0KV | recoverable errors through power cycling allowed |